TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Department of Mechatronics

Chair of Mechatronics Systems

MHK70LT

Mari-Liis Sillat

APPLICATION OF HYPERSPECTRAL IMAGING IN WASTE PAPER SORTING

MSc thesis

HÜPERSPEKTRAALKAAMERATE RAKENDUS PABERJÄÄTMETE SORTEERIMISEL

MSc lõputöö

The author applies for
the academic degree
Master of Science in engineering

Tallinn 2016

AUTHOR’S DECLARATION

I declare that I have written this graduation thesis independently.

These materials have not been submitted for any academic degree.

All the works of other authors used in this thesis have been referenced.

The thesis was completed under .... supervision

“.......”....................201….

Author

.... signature

The thesis complies with the requirements for graduation theses.

“.......”....................201….

Supervisor

.... signature

Accepted for defence.

.... chairman of the defence committee

“.......”....................201… .

.... signature

TUT Department of Mechatronics
Chair of Mechatronics Systems

MASTER'S THESIS TASK

Year 2016 Spring Semester

Student:	Mari-Liis Sillat 132654MAHM
Curricula:	MAHM02/13
Specialty:	Mechatronics
Supervisor:	Early-Stage Researcher Märt Juurma

MASTER'S THESIS TOPIC:

(English)	Application of Hyperspectral Imaging in waste paper sorting
(Estonian)	Hüperspektraalkaamerate rakendus paberjäätmete sorteerimisel

Assignments to be completed and the schedule for their completion:

No.	**Description of the assignment**	**Completion date**
1.	Research of previous work and applications	**03.03.2016**
2.	Planning of experiments	**24.03.2016**
3.	Experiments and data collection	**07.04.2016**
4.	Data analysis	**12.05.2016**
5.	Machine learning	**18.05.2016**

Engineering and economic problems to be solved: Designing and carrying out experiments to acquire significant spectral data of different paper materials. Analyzing data and creating smart algorithms to classify paper types. Creating sorted paper waste with higher economic value by applying algorithms.

Language of the thesis: English

Deadline for submission of the application for defense 16.05.2016

Deadline for submitting the thesis 20.05.2016

Student	Mari-Liis Sillat
Supervisor	Märt Juurma

TABLE OF CONTENTS

FOREWORD

The topic of the thesis was proposed by the Chair of Mechatronics Systems. Introductory work in the same field was done in the same institution [14] but in that case the measurements were conducted by a company outside of the institution. The measurements taken for this thesis were conducted in the Machine Vision laboratory of the Department of Mechanical Engineering. The equipment was also provided by the Department of Mechanical Engineering and Chair of Mechatronics Systems.

The author would like to thank the supervisor Märt Juurma for the valuable insight and help in completing this thesis. The author would also like to thank the Department of Mechanical Engineering and the Chair of Mechatronics Systems for providing the rooms and equipment.

EESSÕNA

Lõputöö teema pakkus välja Mehhatroonikasüsteemide õppetool. Sissejuhatavad tööd samal teemal viidi samuti läbi Mehhatroonikasüsteemide õppetoolis [14], kuid sel juhul telliti mõõtmised instituudist väljastpoolt. Käesoleva lõputöö mõõtmised viidi läbi Mehaanikateaduskonna masinnägemise laboris. Tööks vajalikud seadmed võimaldas samuti Mehaanikateaduskond ja Mehhatroonikasüsteemide õppetool.

Autor soovib tänada töö juhendajat Märt Juurma't väärt nõuannete ja abi eest. Lisaks soovib autor tänada Mehaanikateaduskonda ja Mehhatroonikasüsteemide õppetooli ruumide ja seadmete kasutamise eest.

LIST OF ABBREVIATIONS AND ACRONYMS

HSI – hyperspectral imaging

MSI – multispectral imaging

RGB – Red, Green, Blue

NIR – near-infrared

ROI – region of interest

SG / SGF – Savitsky-Golay (filtering)

SNV – Standard Normal Variate

MSC - Multiplicative Scatter Correction

PCA – principal components analysis

LDA – linear discriminant analysis

MNF transform – minimum noise fraction transform

MDR – major dimension reduction

MR-MIA – multi-resolutional multivariate image analysis

MCR – multivariate curve resolution

PPI method – pixel purity index method

PLS-DA – Partial Least Squares-Discriminant Analysis

SVM – support vector machines (discriminant analysis)

k-NN – k-nearest neighbor (discriminant analysis)

FOV – field of view

N/A – not available

1. INTRODUCTION

The purpose of this thesis was to use hyperspectral cameras to acquire datasets with full spectral information of paper materials in order to find spectral signatures – spectral reflectance patterns inherent only to this material. The data was analyzed to find distinct characteristics of different types of paper, and distinguishable characteristics from other materials. As a result, a method to sort paper waste was proposed which would be beneficial to waste sorting companies. In this work, Resonon Pika II and Resonon Pika NIR hyperspectral cameras were used to gather data of paper products. The data was analyzed to find patterns which would help to discern paper from other materials and – more importantly – distinguish different types of paper based on the quality. This work considered if it was possible to divide paper material into 3-5 groups based on the quality standards set by the companies that buy bulk wastepaper, using a range of spectral information.

The motivation for this work was developing a tool to sort paper waste into different categories based on the quality or grade of the paper in waste sorting companies. Automated sorting of waste is needed in sorting stations where currently the work is done by hand or not done at all. An automated system would be more reliable and repeatable, as well as much faster. This would benefit the waste sorting companies economically because they could sell higher quality paper waste for higher prices. As it was stated in the previous work done in the institute of Mechatronic Systems [14], sorting paper using spectroscopy has many benefits before using RGB cameras and special gloss sensors, such as higher flexibility and cheaper implementation. However, despite the potential of spectroscopy in paper sorting, hyperspectral imaging is not used in waste paper sorting in industrial applications yet, probably due the higher price of hardware. For example, companies like CP Manufacturing and MSS Optical Sorters provide automated paper sorters, but the cameras used are either RGB or wideband NIR, and the sorting machines have only a few paper classes that can be sorted out. A goal of this work is to see if more classes can be separated using the hyperspectral cameras.

As part of the analysis, thorough investigation of the wavelength ranges was conducted to identify if there would be a good combination of a small number of wavelength bands that make discrimination between classes possible. This would allow the production of cameras for this specific task that would not record the whole light spectrum but only the wavelength ranges that are relevant to the application. These cameras would be potentially cheaper and the analysis would be much faster, so it might be possible to produce them on a larger scale.

For the completion of this thesis the following software was used: Resonon's Spectronon Pro for acquiring the hypercubes; MatLab R2015a for the analysis of the data; Eigenvector's PLS_Toolbox for MatLab was used for the PCA analysis and classification models [23]. Additional MatLab toolboxes were used for smaller MatLab tasks: the recursive directory listing toolbox [16]; the Field Mapping Toolbox for 3D matrix filters [17] and MatLab Pattern Recognition and Machine Learning Toolbox [22]. The three additional toolboxes are included in the Matlab Scripts folder of the accompanying source material but the PLS_Toolbox requires a license to run so it must be downloaded by anyone who wishes to run the classification model proposed in this work, and therefore was not added to the source material.

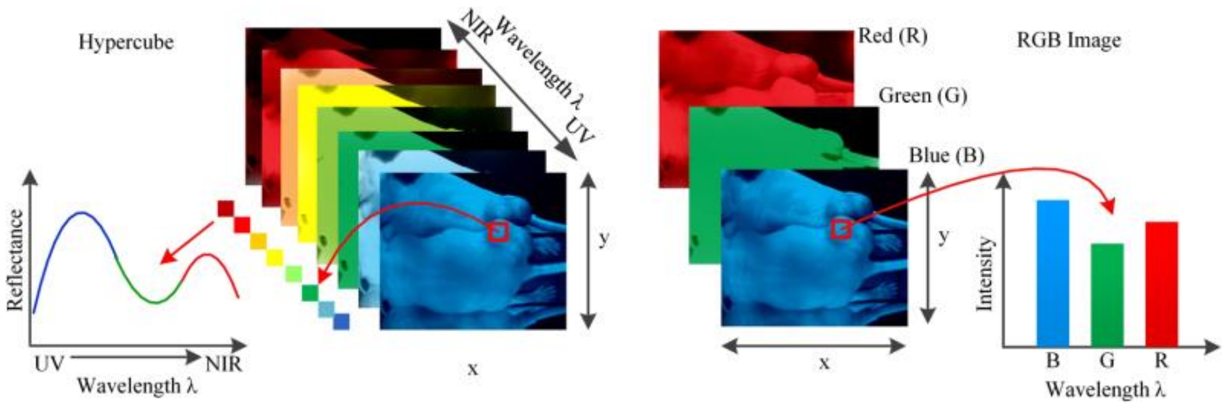
The thesis is organized as follows: Chapter 2 introduces the theoretical basis of hyperspectral imaging, provides an overview of sorting using spectroscopy and describes the requirements of the waste paper sorting system. Chapter 3 provides a detailed explanation on how the measurements were taken with the two hyperspectral cameras. Chapter 4 discusses the possible data analysis and classification methods and describes how they were used in this work and what can be concluded form the gathered data. Chapter 5 gives an overview of the classification models' validation and prediction result using the hypercubes acquired with the NIR camera. Chapters 6 and 7 discuss future usages of the knowledge gained in this work, possible improvements on the classification models and a summary of the work. Chapter 8 is the summary in Estonian. Chapter 9 provides a list of the references used in this work. Appendices 1 and 2 are the datasheets for the Pika II and NIR cameras, respectively. Appendix 3 describes the database of hypercubes that was gathered during the experiments. Appendix 4 is the full table of loadings of the PCA, for the NIR camera, to describe how the relevant variables were selected in Chapter 4. Appendix 5 includes plots of the PCA analysis for the Pika II hypercubes, as presenting them in the body of the work would have been repetitive to the NIR data and also because the Pika II results played a smaller role in this work. Appendix 6 gives an overview of the source data attached to this work. Appendix 7 describes the MatLab scripts that were developed to read, parse, pre-process, manually classify and display the hypercubes and to make them compatible with the toolboxes.

2. THEORETICAL BASIS

This chapter will introduce some of the theory behind hyperspectral imaging. A general overview alongside with a short description of the history and applications will be given. Furthermore, the theory and previous work specifically related to sorting waste or products will be described more thoroughly. Analyzing previous work about sorting paper using hyperspectral imaging, methods and algorithms best suited for the task proposed in this work can be selected.

2.1. Introduction to hyperspectral imaging

The term *hyperspectral imaging* (HSI) consists of three parts. Firstly, *imaging* or *broadband imaging* is the well-known process of taking a photo. Or more specifically – projecting objects in the 3D-space onto a 2D plane within the field of view of the imaging device; by collecting light intensity that has reflected back form objects. *Spectral imaging* refers to simultaneously taking images in many spectral bands – ranges of wavelengths of electromagnetic waves. Spectral imaging can generally be divided into three sub-groups: color imaging, multispectral imaging (MSI) and *hyperspectral imaging* (HSI). The most common form of the aforementioned is color- or RGB-imaging, which is used in all consumer color cameras. In this case color intensity information is saved within three separate wavelength bands. Figure 1.1b illustrates the light intensity of one pixel in the image, measured in three spectral bands.



a.) b.)

Figure 2.1. Hyperspectral imaging compared to RGB-imaging [4]

Multispectral imaging refers to imaging with tens of separate spectral bands and hyperspectral imaging systems use hundreds of spectral bands (Figure 2.1a). According to Kerekes and Schott in [1] there are three characteristics which distinguish HSI from color- and multispectral imaging:

1. As mentioned before, the higher number of co-registered spectral bands. In HSI, the number of bands is much higher (see Figure 2.2) than in MSI.
2. The higher spectral resolution $\frac{\lambda}{\Delta\lambda}$ – center wavelength of the spectral band divided by the width of the band. The smaller the width of the spectral band, the higher the spectral resolution. Thus, the spectral resolution of HSI systems is typically 10 times higher than in MSI systems.
3. The continuous intensity spectrum measured for each pixel within all wavelengths in the range of the device. Comparing the intensity charts of Figure 2.1, it can be seen that the HSI system gives a continuous spectrum for one pixel, whereas the RGB system gives a histogram of widely and irregularly spaced intensities.

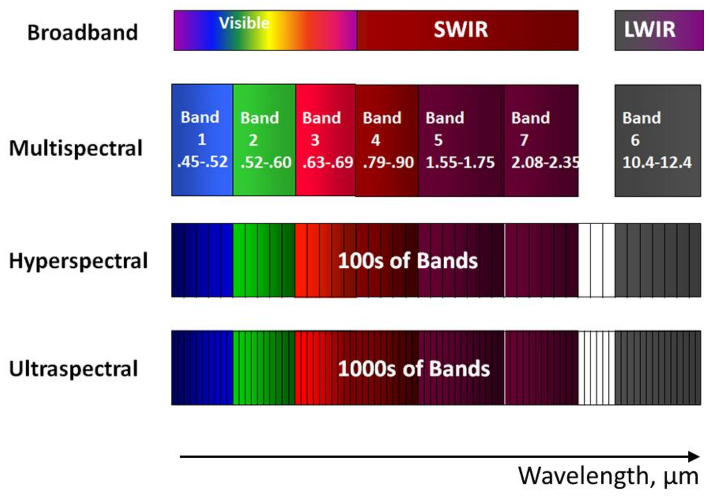


Figure 2.2. Difference between multispectral and hyperspectral data [5]

The HSI cameras used in this work – Resonon Pika II and Resonon Pika NIR – have spectral ranges of 400 – 900 nm (spectral resolution 2,1 nm) and 900 – 1700 nm (spectral resolution 5,5 nm), respectively (for more info on the Resonon HSI cameras, see Appendices 1 and 2). These ranges fall into the Near Infrared (NIR) category of spectroscopy. In comparison, a standard RGB-camera has a spectral range of visible wavelengths 400 – 700 nm and a typical spectral resolution of about 100 nm.

As a result of HSI a three-dimensional matrix is recorded, containing spatial and spectral data. This matrix is called a *hypercube* and a simplification of it can be seen on Figure 2.1a. For each pixel (a point in space) there exists a one-dimensional spectrum of wavelength versus reflectance or intensity. The spectrum – spectral signature or spectral fingerprint – depends on the state and material of the object and is in fact characteristic to the material. Therefore, the data gained from HSI is mainly used to identify and characterize features such as materials in images. It has been shown in experiments [1] that most characteristics of various materials become present within wavelengths of 400 to 2500 nm and spectral bands of 5 to 20 nm, which is within the technical capabilities of HSI, therefore making it an extremely appealing field of study for remote material identification.

2.2. Applications of hyperspectral imaging

Hyperspectral imaging first became viable in the 1970s when optical detector array technology advanced to a level where it became possible to collect spectrally continuous and spatially correlated data with 2D-sensors. Since then, HSI systems have been used in many different fields of study. Remote sensing of the Earth's surface was one of the very first applications for HSI systems. The goal in the first applications was to find minerals and oils with remote non-invasive sensing. As a result of years of research of the Earth's surface, two publicly accessible spectral libraries have been collected: the ASTER Spectral Library and the USGS Spectral Library which contain the spectral information of the most common minerals, rocks, soils etc. found on our planet's surface [2]. Unfortunately these do not contain a sub-library of paper materials which would be very useful in the context of this thesis. Nevertheless, nowadays HSI is applied widely in agriculture and in environmental as well as military applications.

Another use for HSI systems is in medical imaging. HSI makes it possible to analyze human tissue non-invasively and at the same time provide large amounts of data. The more common areas in medicine where HSI systems are used are diagnosing cancer, cardiac disease, diabetes, retinal disease and performing image-guided surgery. In addition to recording reflected light from tissues, medical HSI systems can implement fluorescence and transmission modes of HSI. Those mean injecting fluorescent additives to the inspected tissue (skin cancer, for example) or placing a light source behind the tissue so the light is transmitted through the tissue (for example, in microscopic inspection), respectively [4].

Recently HSI systems have been taken into use in the industry as well. The frontrunners in applying HSI systems are food processing companies. The goal in food processing is reliably and in-line at full production volumes sorting out the foreign bodies and defective products. HSI systems are very widely used in the nut industry as well as the potato industry because the production values are very large and the foreign material frequent and nearly indistinguishable from the product with traditional sorting devices like machine vision and laser sensors. The higher cost of the HSI sorting system is justified due to the improved product quality, low false-positive rates and the HSI systems' ability to handle high loads.

What is common among all of the above-mentioned applications is the fact that objects in the image are classified and grouped on the basis of the collected spectral fingerprints (see Chapter 2.3.1). The classification is achieved with a number of carefully chosen data processing methods and algorithms. The suitable methods for sorting paper waste are discussed in Chapters 2.4.3 and 2.4.4.

2.3. Terminology related to hyperspectral imaging

The HSI hypercube contains a vast amount of data. To be able to analyze and use the captured information, one must understand the physical phenomena that cause the readings on the optical sensors. The following chapters describe the most important terminology related to HSI. It is important to note that this chapter only includes the terminology relevant to sorting applications in controlled environments, and terms such as atmospheric and adjacency effects – relevant for remote sensing – are not discussed here.

2.3.1. Spectral radiance and reflectance

Spectral radiance is a directional quantity that describes the transfer of optical energy in a specific direction. It is the radiant flux received by the surface of the hyperspectral sensor, per unit of solid angle per unit of projected area per unit of frequency or of wavelength. The SI-unit of spectral radiance is $\frac{W}{sr \cdot m^2 \cdot Hz}$, which is the radiance received by the surface per unit of frequency. For HSI systems the unit of spectral radiance is more commonly the radiance received by the surface per unit of wavelength $\frac{W}{sr \cdot m^2 \cdot nm}$.

Spectral reflectance is measured as the spectral radiance reflected from the surface of the target material, as a ratio of reflected energy to the incident optical energy as a function of wavelength. The changes in reflectance depending on the wavelength are due to the physiological and chemical structure of the materials. Spectral reflectance of materials is the basis of classifying them due to the fact that materials absorb or scatter light at different wavelengths in characteristic manners, producing a *spectral fingerprint* (see Figure 2.3).

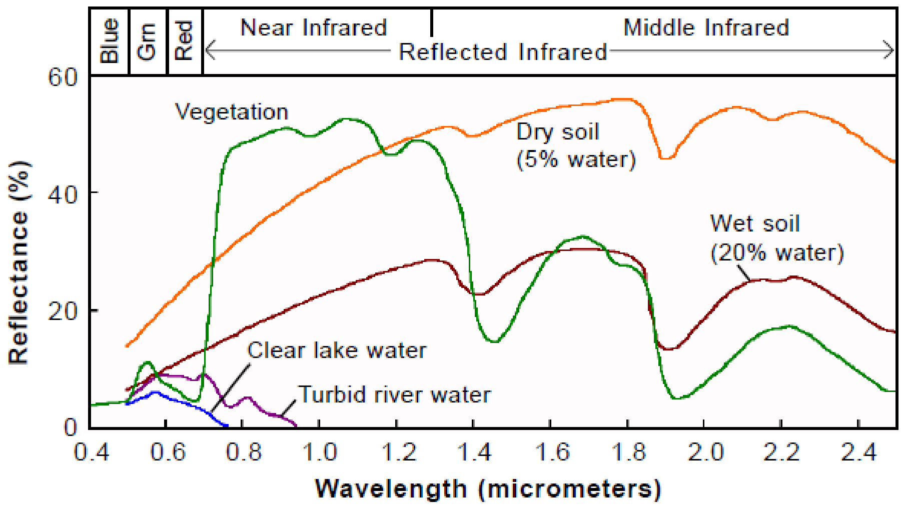


Figure 2.3. Spectral reflectance of common materials [2]

Even though the spectral fingerprint is extremely important for classification, it is an oversimplification to say that it is consistent for materials in all environments and cases. There are numerous effects which can lead to variations in the spectral shape of the fingerprint, such as angle of illumination, angle of view, additives in the chemical composition of the material, age and exposure to the environment. However, these variations are mostly small in magnitude and spectrally correlated to the data, so further processing makes it possible to take these variables into account. Moreover, in the application of waste paper sorting in a waste treatment facility it is possible to create a fairly controlled environment to make the data more consistent.

2.3.2. Spatial resolution and spectra mixing

The term *spatial resolution* is used in every form of imaging and refers to the size of the area represented on the image by one pixel. Spatial resolution is directly related to the distance of the object from the imaging system's sensor as well as to the optics used and the parameters of the sensor. For example, in remote sensing the spatial resolution can span from meters to kilometers, depending on the altitude from which the imaging is done.

Spectra mixing occurs when within one imaged pixel there are several materials which have different spectral fingerprints. In the context of waste paper materials, common non-paper materials that could be present within one pixel are ink, paint, all kinds of dirt, plastics, metals (paperclips, staples etc.) and cloth. Each material's contribution to the mixed spectra is directly proportional to its area or composition percentage within the pixel, therefore the mixed spectra is a linear function of the spectra of its components. However, mixing spectra is directly related to the spatial resolution of the system, and the non-paper materials will be fairly large compared to the spatial resolution, then it can be said that these objects can be detected as objects within the frame. If the non-paper materials cause spectral mixing, then linear unmixing can be used. Linear unmixing is a process where it is assumed that the scene only includes a small number of different materials with quite constant spectral fingerprints or spectral *endmembers*. The variability in spectra is caused by mixing the spectra of the endmembers.

2.4. Fundamentals of hyperspectral data collection and analysis

This chapter summarizes the data acquisition methods and data analysis algorithms used in previous work related to material and/or product sorting in an industrial environment. Table 2.1 gives an overview of the research. From the Table 2.1 it can be seen which steps have led to successful classifications of materials and therefore what methods are needed and proven to work. Based on research, the experimental setup of this thesis has been put together.

Sub-chapters 2.4.2 – 2.4.4 divide the processing and analysis into three steps – calibration, pre-processing and calibration – and describe each of the methods and algorithms in Table 2.1 more thoroughly and provide the reasoning why they are relevant in the waste paper sorting application. The results of the research are presented in Table 2.2 in Chapter 2.5, where the relevant information for this work has been organized and described.

Table 2.1. Overview of previous work

Ref.	**Sorted Material**	**Camera (range / spectral resolution)**	**Experiment Setup**	**Calibration**	**Software**	**Used Algorithms**	**Results**
[3]	4 classes of plastics + background	Headwall 1002A-00371 (1009-1694 nm / 4.85 nm). Line mapping (320 px/line); 14-bit CameraLink	Camera perpendicular to the sample; Illuminated w/ diffuse white light at 45^o in respect to the sample; Reflectance mode	Dark current and 99% reflectance calibration w/ Spectralon plate; Correction of the hypercube performed by the software	MatLab toolboxes: HYPER-Tools, PLS_Toolbox, Imge Processing Toolbox	K-means clustering, SNV, Savitzky-Golay, mean centering PCA, MCR, PLS-DA classification, statistical assessment	Successful classification and sub-classification of 5 categories of plastics
[7]	Horticultural crops	N/A	Same as [3]	Dark current and 99% reflectance calibration	N/A	SNV/MSC, PCA, MNF, PPI, MR-MIA, PLS classification	Overview of all steps in HSI process
[8]	Hazelnut	XEVA-FPA-2.5-320 (850-2500 nm/ 7,5 nm), 320x256 px. Two exposure times used to get a HDR image to compensate for the lower sensitivity in the 1800-2500 nm	Camera perpendicular to the samples; 4x20 W halogen spotlights (DC 2800 K) w/ DC voltage source to avoid output fluctuations at 30^o and 60^o on both sides of the sample	Dark current and 99% reflectance calibration w/ 75% Spectralon plate, repeated after every 20 scans	LabVIEW, MatLab, PLS_Toolbox	Binary masks, same preprocessing as [3] and [7] PLS-DA classification, additional spatial information used to improve classification	Successful sorting into 4 classes
[9]	Metal alloys	VNIR camera (320-1030 nm / 6,25 nm) Specim SWIR camera (1000-2500	Halogen lamp array, light reflected from a extruded elliptical reflector	Same as above	MatLab PerClass	Smile correction, PCA and LDA (LDA proved to be better). Random	Metal allow classification

		nm range)				forest algorithm for classification was best	
[10]	Nonferrous materials	Specim PHL Fast10 CL (384-1008 nm)	Halogen and near-UV lights, diffuse covers	Same as above	N/A	Binary classification for background; PCA compared with Spectral Fuzzy Sets; Neighborhood Fuzzy Histograms, Region Merging	Fuzzy sets perform better than PCA
[11]	Waste metals	PHF Fast10 camera (400 – 1000 nm / 1 nm), 1024 × 1024 px, Fore OL10 lens	Halogen and LED lights, parabolic reflectors, range of 400 – 1000 nm	N/A	LabVIEW	Uses [10]	Full sorting line, separates nonferrous materials into 6 classes
[12]	Waste cellulose-based materials	ImSpector N17 (900 – 1700 nm / 13 nm), 288 px/line, 100 fps NIR-optimized lens	Halogen lamps, 900 – 1700 nm, radiation intensity >70% in that range, material illuminated at a 70^o angle	Adjusting the camera and spectograph And same as above	N/A	PCA and then LDA on top; k-means clustering	Classifying 4 groups of cellulose-based materials
[14]	Paper	Imspector V10 (400 – 1000 nm / 6,8 nm) and Imspector N17E (900 – 1700 nm / 5 nm)	Wide spectrum halogen light unit	Same as above	LabVIEW	N/A	It is possible to separate white paper from waste paper stream

2.4.1. Structure of hyperspectral images

Before applying algorithms to the data, the structure of the hypercube must be understood. The hypercube is a 3D dataset $\underline{\mathbf{H}}(x \times y \times \lambda)$ where *x* and *y* represent spatial information and λ represents spectral information. The NIR camera, for example produces hypercubes that have 320 pixels of width (*y*-dimension) and 44 usable wavelength bands as λ (while using binning with a window size 3). The *x*-dimension is arbitrarily chosen by the user and it depends on the field of view, integration time of the sensor and angular velocity of the camera head. The Pika II has higher resolution with 640 pixels in the y-dimension and 59 wavelength bands (also using windowing 3). The cameras have bit depth of 14 and 12 bits, so each layer in the hypercube along the λ-axis is an image of that bit depth at that wavelength band. For every point (x_i; y_i) there corresponds a spectra λ_i.

2.4.2. Calibration and image correction

Corrections for gain, dark current offset and variable integration time during image processing need to be taken into account when analyzing hyperspectral data. Causes for the aforementioned phenomena are bad quality equipment, heterogeneous lighting and inadequate resolution of analog-to-digital conversion. It is essential to choose equipment of good quality and that suits the application but it is important to note that it is impossible to get good quality data just with good quality equipment. The data collected from the camera sensor represents only intensity counts at the specific sensors but not the actual reflectance values. To get the actual reflectance and absorbance values the HSI system must be calibrated.

The camera sensor arrays have wavelength-dependent dark current – a small flow of electrons at the sensor even when no photons reach the sensor. To take this current into account, a dark current background image *D* must be recorded. The same applies with the opposite situation – a reference image *W* is acquired from a sample with nearly 100% reflectance (99% standard). Using these values and the reflectance intensity count from the sample, *S*, a calibrated percentage of reflectance, *R*, can be calculated using the formula (2.1) [7]:

$$R = \frac{S - D}{W - D} \qquad (2.1) \qquad\qquad A = \log\frac{1}{R} = -\log\frac{S - D}{W - D} \qquad (2.2)$$

where *R* – percentage of reflectance of the sample
S – measured reflectance intensity
D – dark current reference
W – 99% reflectance reference

Using the same values, absorbance values of samples can also be calculated using the formula (2.2). This equation is linear – because only one reference standard value is used – and is called the one-point calibration. Since a 100% reflectance standard is difficult to obtain then it must be considered that still minor errors in reflectance *R* and absorbance *A* can be present. To detect nonlinearities 2-, 5-, 50-, 75- and 99% reflectance standards are used. To obtain properly calibrated data, the sensor temperature must be constant because noise and the cutoff wavelength are temperature-sensitive.

According to [12] it is extremely important to adjust the camera so that the spectral dimension is precisely projected in a vertical direction onto the sensor. This is achieved using a wavelength calibration lamp. These lamps produce peaks at certain wavelengths, but these peaks do not register as sharp lines in the image but as wide blurry lines. It is not possible to physically adjust the camera so that the lines on the image become sharp. Therefore sub-pixeling must be used to calculate the positions of these maximums. The procedure consists of five steps [13]:

1. Estimate the spectral center position of each peak for every spatial pixel across the spatial axis
2. Fit a second-order polynomial through the estimated peak centers
3. Estimate the spectral peak positions for the first and the last spatial pixel column by use of the 2nd order polynomial
4. Estimate the difference between the two peak positions
5. Rotate the camera until the difference is zero

2.4.3. Pre-processing

To decrease inspection time and the amount of data to be analyzed, *spatial pre-processing* is used to mark Regions Of Interest (ROIs) on the image. For example this can be done with a binary masking matrix, where values 1s denote areas to include in the inspection and 0s areas to discard. In the case of waste paper sorting spatial masking could only be used during training and/or data gathering procedures and not during in-line inspection due to the unforeseeable location of the waste.

Spectral pre-processing is performed to increase the spectral differences between the background and the inspected objects. In reference [3] k-means clustering shows great results in removing the background from the various materials. The clustering is done on the first derivative of the data because derivatives increase the differences between spectra as well as minimize spectral noise and artifacts. Subsequently, the Savitzky-Golay (SG) filter is used for smoothing and de-noising of the data. Moreover, Standard Normal Variate (SNV) filter or Multiplicative Scatter Correction (MSC) filter is used in NIR spectroscopy to normalize the data and to minimize the effects of light scattering. These two aforementioned filters generally give similar results, but occasionally the results differ and the comparison of the two may give interesting insight. It is important to note that spectral pre-processing might cause extra noise or loss of information if not used correctly and therefore the filters must be applied with care.

Hyperspectral data cubes consist of vast amounts of spatial and spectral data, a part of which is either corrupted or simply nonessential to the task at hand. To reduce the amount of data and storage space required, principal components analysis (PCA), minimum noise fraction (MNF) transform and linear discriminant analysis (LDA) methods are used. These methods reduce the dimensions of the data when at the same time preserving all of the critical data of the samples, especially LDA. Furthermore, the pixel purity index (PPI) methods can be used the further reduce the spatial dimensions. 3D wavelet transformations are also used in size reduction but based on [7], the multiresolutional multivariate image analysis (MR-MIA) performs better than any of the five Daubechies family wavelet types in the case of hyperspectral data. These methods help separate the data and analyze if the samples are separable into classes, they are not classification methods. They also provide a visual way of confirming the separability of the classes.

2.4.4. Classification methods

The sample classification is achieved with either supervised or unsupervised classification methods. Most of the classifiers used in previous research were fairly simple, the simplest method was k-nearest neighbor clustering. K-nearest neighbor clustering is a non-parametric supervised classification technique which is relatively simple but also has been proven to be effective in many cases [12]. The technique is based on the assumption that pixels that are close to each other in the feature space are also likely to belong into the same class. This method measures the distance between an observed pixel vectors and all the other pixels. The pixel belongs to the class where the distance is smallest, but above a set threshold. By selecting the number of likely neighbors, the separation boundaries can be very flexible and give great results. Another widely used method in classification was Partial Least Squares-

Discriminant Analysis (PLS-DA), used with great results in [3] and [8]. It is a supervised classification method that uses the partial least squares method to determine if the sample belongs into a class. For this method it is necessary to create the class calibration matrices: one with the samples of data and another with binary data of classification.

2.5. Plans for experiments with the HSI system

As a result of the previous research, a plan of how to conduct experiments with the hyperspectral cameras was composed. The following Tables 2.2 – 2.5 include the methods and algorithms that should be applied on the HSI system and on the acquired data, presented in order by which they should be applied.

Table 2.2 Experimental setup of the camera, lighting and background

Device	**Detail**	**Notes**
Camera	Camera position	Sensor perpendicular to the test subjects
	Lens	Optimized for NIR
	Acquisition	Test with different exposure times to suit the sensitivity of the sensor across the wavelength scale
	Filtering	Order blocking filter is needed if $\lambda_2 / \lambda_1 > 2$, where λ_1 and λ_2 are the minimum and maximum recordable wavelength of the sensor, respectively [12]
Lighting	Lighting source choices : • **tungsten halogen** (durable, stable, 400-2500 nm light, but emits a lot of heat) • quartz halogen • LED • tunable lasers • heated xenon lamps	Characteristics should be [7]: • Homogeneous illumination over FOV • Short pulses <10 fs • Intense polychromatic light • Polarized light (know Stokes parameters) • Excellent transmission through samples • Controlled reflection from sample • No radiation damage to samples
Background	Based on preliminary tests with the Pika II and NIR cameras, the most suitable background turned out to be the one with the minimal reflectance.	Based on [15], the lowest reflectivity of a common non-metal material is black felt.
Software	Resonon’s Sprectronon Pro MatLab	For Spectronon Pro, check if slope and intercept values are set as with calibration sheet

In Table 2.3, the suggestions for calibrating the system are brought out. However, the steps 1-3 are done by the Spectronon Pro software and step 4 is unnecessary in this application due to the fact that alignment has been done by the manufacturer of the cameras. Regardless, calibration is a necessary step that must be repeated after a number of scans. Table 2.4 shows the pre-processing steps and Table 2.5 describes the classification options.

Table 2.3. Plan for calibrating the hyperspectral cameras

Order No.	**Method**	**Notes**
1.	Acquire dark current image	Black non-reflective cap on lens
2.	Acquire reflectance image	With reference plate
3.	Apply equations (2.1) and (2.2) on acquired data	Should be repeated after a number of scans (for example, 20)
4.	Calibrating the alignment of the axis of the spectograph and the camera sensor, using a wavelength calibration lamp	Might not be required Based on [13], pages 69-72

Table 2.4. Hyperspectral data pre-processing plan

Order No.	**Filter/method**				**Goal**	**Notes**	
1.	Binary mask				Separate background from objects	Easier than k-means	
	K-means clustering on derivative spectra					Try 2-5 clusters	
2.	Savitzky-Golay filtering		Wavelets (Daubechies family of orthonormal wavelets)		Smoothing and denoising	Find optimal window size	Try if SG has no result
3.	SNV		MSC		Normalize data, minimize scattering	Choose best of two	
4.	MNF	PCA	PPI	MR-MIA	Reduce dimensions	Less relevant for sorting	
5.	PCA		LDA		Find the Principal Components where the differences between classes are biggest; data decorrelation	Gives clues about what classification algorithms to use and gives a visual overview of the different classes	
6.	Fuzzy Sets, Neighborhood fuzzy Histograms, Region Merging				If PCA or LDA prove unsuccessful in decorrelating the data	Fuzzy sets and region merging have given best results in [10] and [11]	

Table 2.5 Classification plan

Order No.	Method	Description	Notes
1.	Non-normal distribution of the residuals	Check the non-linearity of the models to make sure if the problem can be solved with linear classification models	If correlation between the spectra and target property is linear, use PLS-DA
2.	Non-linear relationship in the prediction		Else, use non-linear models, like support vector machines (SVM)
3.	PLS-DA	Main classification process	Find best suited classification method
	k-NN		
	SVM		
4.	Statistical assessment of the model	CAL, CV and PRED methods	Enables to assess the prediction performance
5.	Region merging and re-classification	Merge the different classes of connected regions (within one sample) into one	Based on minimization of the matching cost criterion

These tables were used as a reference in conducting the experiments. The experiments in are described in Chapter 3 and the data processing and analysis is described in Chapter 4.

2.6. Classification of waste paper

In recycling waste paper it is important to sort paper into classes because the purer the material the more effectively it can be used as a raw material in paper production. In general, the paper is divided into two groups: paper with bleached fibers and paper with non-bleached fibers. Since only bleached paper can be used in producing white paper then it must be sorted out from the waste paper stream. Furthermore, fiber length also plays an important part in recycling: short-fibred cardboard, for example, cannot be used in producing white paper, and therefore must be separated from white paper.

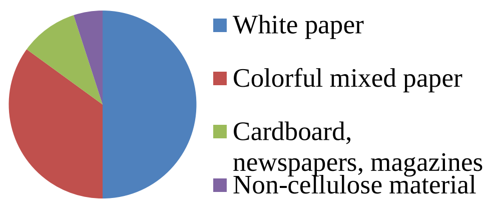


Figure 2.4 Approximate concetration of paper classes in the waste paper stream

The main principles of sorting waste paper come from European Union directives [18] as well as the needs of the commissioning company. The goal is to sort out white bleached paper, cardboard, newspaper and others from the waste paper stream. The stream can also include non-cellulose materials in the form of paperclips, plastic binders etc. The concentration of paper classes in the waste paper stream is shown in Figure 2.4. The desired classes and subclasses are described in Table 2.6. According to the information given by the Chair of Mechatronics Systems and based on the recycling process, the classes marked with a gray background in Table 2.6 are higher priority and more important to sort out from the waste paper stream. Identifying clean pieces of white bleached paper is of the highest priority because it has the highest market value.

Based on the information on classification and similarly to the database in [12], a library of paper and cardboard materials was carefully selected. Appendix 3 shows the full library, in which there are over 500 samples of paper, carton and cardboards, as well as samples of paper in plastic covers and covered in tape. In this library all samples were classified according to Table 2.6 and to their physical appearance. However, the white papers with print were marked only as white papers and unlike in Table 2.6 where the percentage of the coverage was given. The percentage was calculated by the proposed method in this thesis and would be given as additional information on the output.

Table 2.6 Classes of cellulose-based materials

Class	White bleached paper (WP)		News-paper (NP)	Magazine (MG)	Cardboard (incl. carton) (CB)	Office supply	Copy paper	Non-cellulose materials
Subclasses	Clean paper				Cardboard	Post-its		
	Up to 10%	Print			Carton	Checks		
	Up to 50%					Colorful p.		
	Over 50%					Etc.		

2.7. Summary of theoretical basis

In Chapter 2, the theoretical terms and real-life applications of hyperspectral imaging were discussed. A research of previous work was conducted, based on which the plan for the tests was put together. This plan includes suggestions for the setup of the camera as well as a list of preprocessing steps to take and what previously successful classification methods to use. Based on the European Union directives, the needs of the client and the recycling process, a library of papers, cardboards and cartons was carefully collected and classified. This library was used in the experiments described in Chapter 3.

3. DATA ACQUISITION

In this chapter the process of the experiments is described, from the setup to the data acquisition. The Chapter 3.1 describes how the camera and stage were set up and Chapter 3.2 describes the calibration process as well as how to images were acquired.

3.1. Camera and stage setup

The test setup was based on Table 2.2. This chapter describes the actual setup in detail. Figure 3.1 shows the setup with the NIR camera. The setup with the Pika II was basically the same, differing only in the working distance and lens selection.

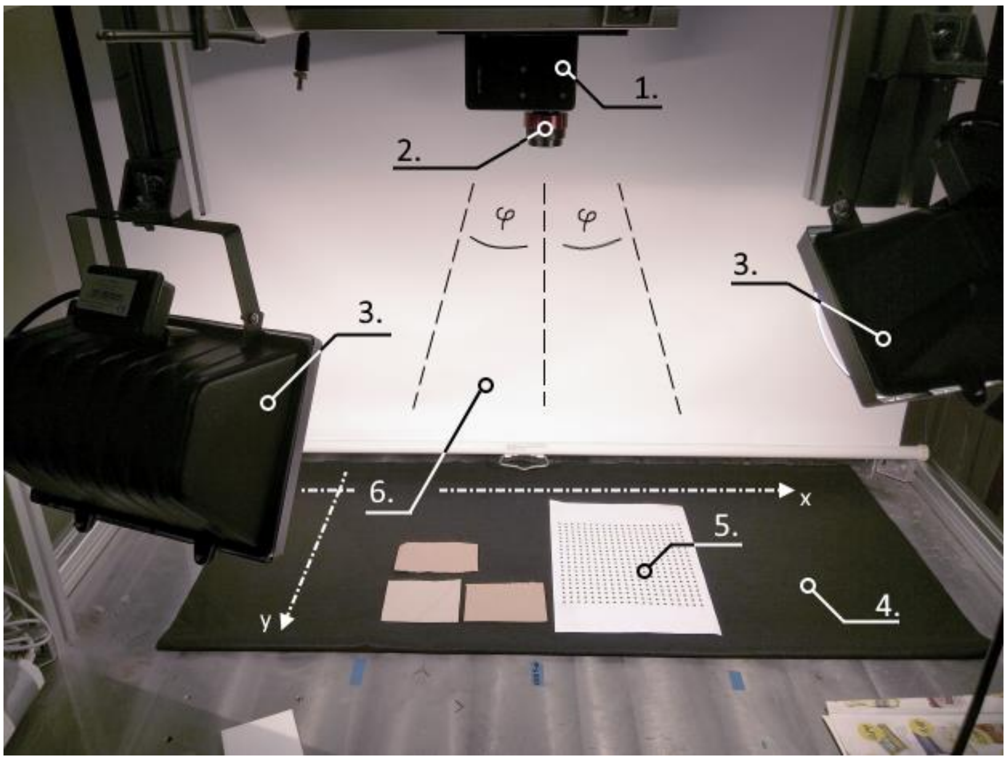


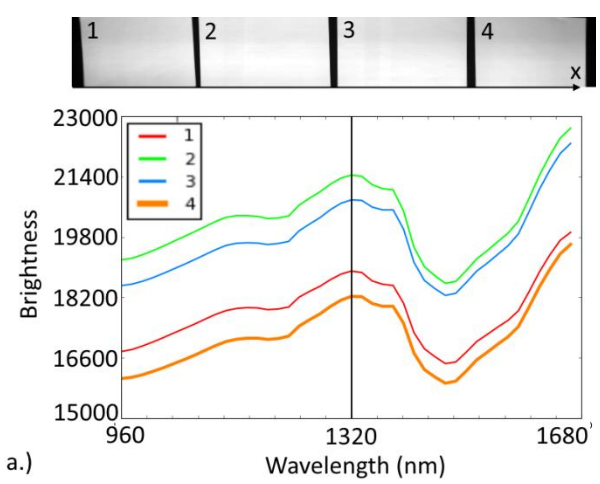
Figure 3.1 Front view of the test setup. 1. Resonon NIR camera; 2. Stingray Opticts lens; 3. Halogen lamps; 4. Black felt background; 5. Test subjects; 6. White diffuse reflective background.

Both cameras were linescan cameras and therefore needed either an actuator for the samples or the camera itself to accumulate a 2D image. The camera was attached to a stepper motor with which it was possible to change the angle φ of the camera. Ideally the sensor would have been at $\varphi = 0$ at all times (therefore the sensor being perpendicular to the samples) and the samples would have moved on a linear actuator below the camera, but unfortunately that was not possible. Nevertheless the measurements taken were suitable for the task due to the uniform lighting (see ahead) and the fact that the distance from the sensor changed minimally and it did not affect the spectral response. The lenses and image acquisition or integration times are given in Table 3.1. The lenses were provided by the camera manufacturer and were suitable for the respective wavelength ranges. The working distance was chosen so that it would be comfortable to lay out the samples in front of the camera. The mount of the camera was at a constant height but due to the different dimensions of the Pika II and the NIR, the sensors were at a varying distance from the samples. Also, due to the differences in pixel size and lens focal length, the field of view (FOV) along the y-axis varied between the cameras. The field of view along the x-axis was dependent on the maximum angle of φ which was chosen to be about 25-30^o to keep the change in working distance to a minimum.

Table 3.1. The camera lenses, field of view and integration times' specification

Detail		**Resonon Pika II**	**Resonon NIR**
Lens	Name	Schneider Xenoplan 1.4/23-0902	Stingray Optics SR0907-184
	Focal length	23 mm	25 mm
	Used aperture setting	f11	f_{max}
Field of view	Pixel size (# of px)	7,4 µm (640)	30 µm (320)
	Along x-axis FOV_x	About 450 mm	
	Along y-axis FOV_y	150 mm	210 mm
Working distance	Minimum (φ = 0)	495 mm	600 mm

The lighting choice for the tests was also based on Table 2.2, previous research and the Resonon reseller. Two industrial halogen lights were used (as seen on Figure 3.1, 3). Halogen lights output light in the wavelength range of 400-2500 nm which was perfectly suitable for the task. The field of view depended on how many lines the camera scanned. This was chosen to be so that the entire field of view would have been illuminated as uniformly as possible. Therefore, a field of view along the x-axis of about 450 mm was chosen, because the two industrial lamps could not illuminate a larger area uniformly enough (see Figure 3.2). In addition to the two lamps, a white diffuse reflective screen (marked as 6. on Figure 3.1) was positioned on the opposite side of the samples from the lamps. This would provide a more uniform and diffuse lighting of the samples, as light from the lamps would reflect on the samples furthest away from the lamps. The light sources were not polarized, however a linear polarizer was used for the Pika II.For the NIR measurements, excess reflection was not a problem and a polarizer was not used.



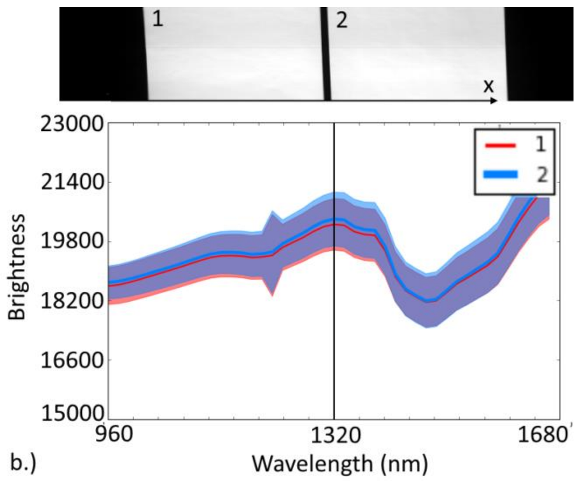


Figure 3.2 Images at 1320 nm. a.) Illumination of FOV_x = 900 mm, without white diffuse screen. Samples 1 and 4 are much less illuminated than 2 and 3. b.) FOV_x = 450 mm, with white diffuse screen. Also shown is the standard deviation of the samples 1 and 2.

As stated in Table 2.2, the background was chosen so that it reflects the least amount of light. Various backgrounds with different reflectances were tested, the background materials with the highest reflectance affected the spectral responses of the samples the most. Therefore, based on [15], common black felt (with a thickness of about 1 mm) was used as a background for a very low reflectance fingerprint throughout all the wavelength range (see Figures 3.3 and 3.4).

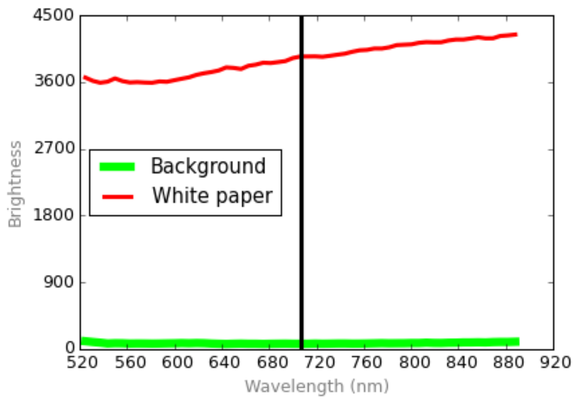


Figure 3.3 Mean spectra of black felt background compared to the mean spectra of a random sample of white paper. Pika II spectra.

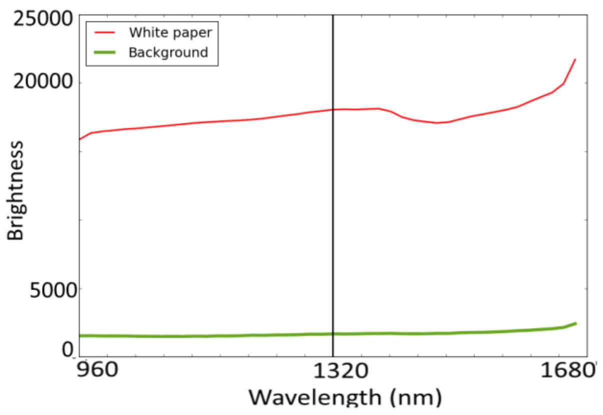


Figure 3.4 Mean spectra of black felt background compared to the mean spectra of a random sample of white paper. NIR spectra.

Choosing the acquisition parameters – such as integration time, framerate, gain – and stepper motor settings – angular velocity and acceleration – was done experimentally using the

Spectronon Pro Software. Table 3.2 shows these parameters for the respective cameras. The Pika II sensor needed a lot of light and therefore the integration time was quite long, forcing the framerate to be very low. The stepper motor was operating at its minimal angular velocity, yet still it was not possible to retain the spatial dimensions, therefore on the Pika II images the x-dimension has less spatial information than the y-dimension (see Figure 3.5 b.). The NIR sensor was much more responsive to light and so in that case it was possible to retain the x-dimension spatial information as well (see Figure 3.5 a.). The spatial distortions caused by the change in the working distance are shown on Figure 3.6. As seen in Table 3.1, the aperture settings of the lenses are at the maximum, so the aperture is very small. This was chosen also because of the change in the working distance, to increase the depth of field and doing so, keep the samples in focus (Figure 3.6, all samples were in focus). The software used sample binning (bin number = 3) so that the spectral responses of three adjacent spectral bands were combined into one data point. This was used to reduce the size of the acquired data and increase the signal-to-noise ratio.

Table 3.2. Final camera acquisition parameters

Parameter		Resonon Pika II	Resonon NIR
Acquisition parameters	Integration time (ms)	32.8	11101
	Framerate (Hz)	30	30
	Gain	0.0	N/A
Stepper motor parameters	Angular velocity (deg/s)	1.0	1.6
	Acceleration (deg/s^2)	15	15

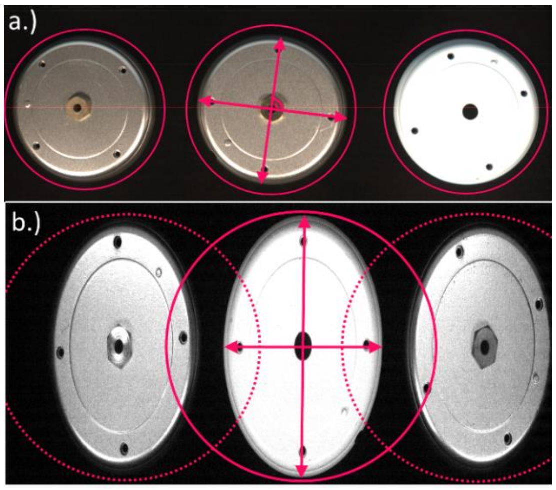


Figure 3.5 Image compression along the x-axis. Pink lines represent full circles, objects are circular. a.) Image with the NIR camera, distortions are minimal. b.) Image with the Pika II camera, distortions are visible

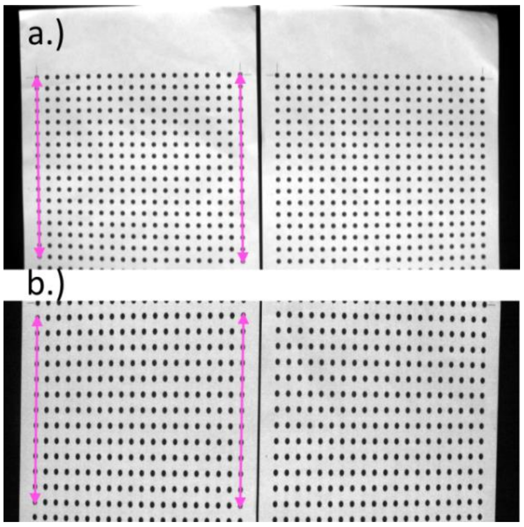


Figure 3.6 Image compression along the y-axis due to the varying working distance. a.) Nir camera: compression 4,0 %. b.) Pika II camera: compression 3,3 %

Interestingly, the wavelength ranges for the cameras had to be adjusted because at the limits of the working range given in the datasheets the data was very noisy or completely random white noise. The hypercubes of both cameras were cropped: NIR data was usable from 960 nm to 1663 nm and Pika II data from 520 nm to the maximum 888 nm.

The setup described in this chapter remained constant throughout all the tests with the cameras. The camera setup, lighting and background remained the same with both cameras, the lenses; field of view and acquisition parameters changed to suit the needs of the respective sensors. The chapter 3.2 describes the repeated calibration process and the acquisition of images in more detail.

3.2. Calibration and image acquisition

The calibration process was described in Table 2.3. Based on previous research and the suggestions of the cameras' manufacturer, the calibration process should have been repeated after a constant number of scans, for example 20 scans. However, in this case the calibration process was repeated for each group (class) of the samples (see Appendix 3). On average, this meant recalibration after every 5-10 scans, but the time between each recalibration could be several hours so the temperature of the sensor could have changed drastically. The calibration process was fairly simple, thanks to the Spectronon Pro software, which did the calculations (using Equation 2.1) automatically, after a two-step calibration process:

- Dark image recording with a lens cap on the lens to eliminate dark current effects
- White reflectance image recording with a Spectronon plate of known reflectance.

As mentioned before, the image acquisition was done in groups, the samples manually classified and the samples given codes and numbers by which they could later be retraced, if necessary. For each group, a text file containing the classification info of each single sample was composed, named *Scan Info.txt*. Figures 3.8 and 3.9 show examples of acquired hyperspectral images, rendered from one wavelength band.

The samples were positioned on the background material so that none of the samples' edges would be touching. This would make the sample separation process easier in this work so focus could be put on the spectral analysis of the data and not machine vision analysis for separating blobs (samples) on the image.

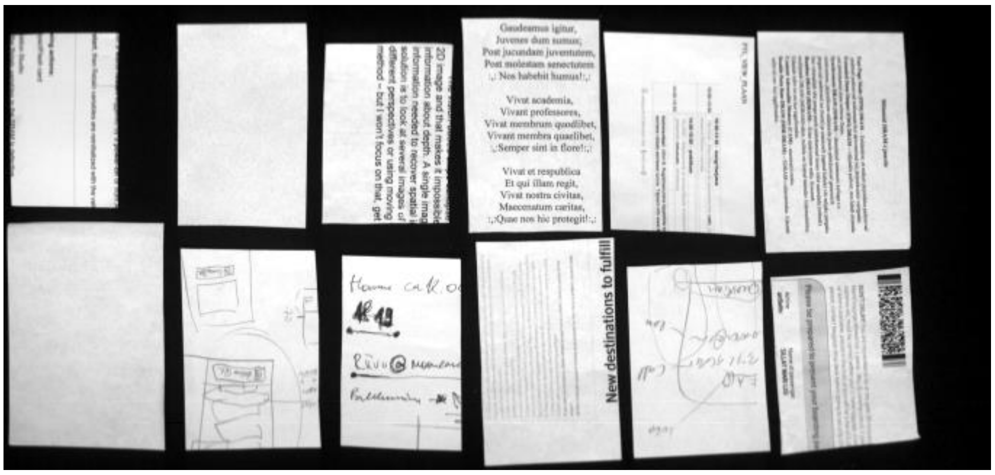


Figure 3.7 Example of NIR image at wavelength 1200 nm.

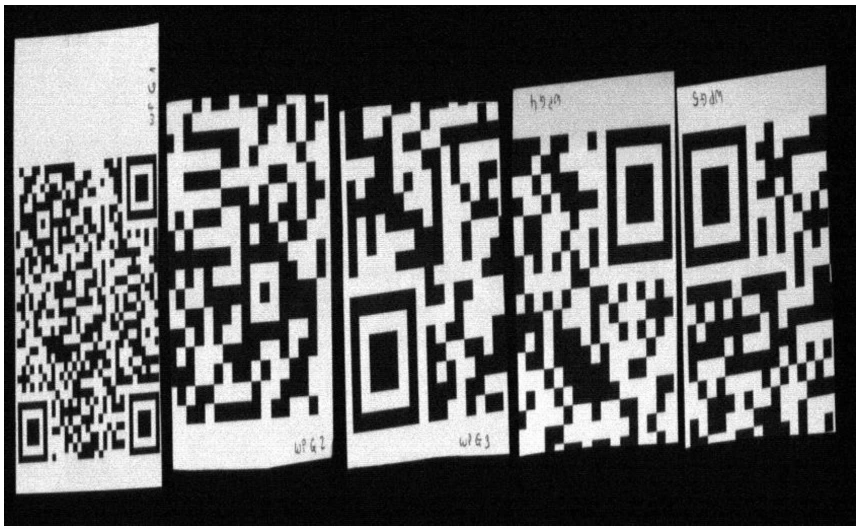


Figure 3.8 Example of Pika II image, at wavelength 643 nm.

4. DATA ANALYSIS

This chapter describes the process of analyzing the vast amount of data acquired in Chapter 3. The steps applied are in the same order as in Table 2.4. This chapter is divided into three parts: data pre-processing, the principal components analysis and the classification. For simplicity, the analysis given below is mostly based on the NIR spectra. The results of the Pika II analysis are summarized in Chapter 4.2.1.

4.1. Data pre-processing

The goal of the preprocessing is to separate the samples from the background, smooth the data and remove noise, normalize the data and minimize scattering and lastly, separate the non-essential data from the essential. To do all of this, a huge number of methods and algorithms have been proposed by mathematicians and statisticians but in this thesis the successful methods used in previous similar applications have been used.

4.1.1. Initial analysis

To get an overall estimation of how the spectra of different paper classes would look like and if some conclusions of the data can be made by eye, some examples of data were taken and plotted (Figure 4.1 for NIR data and Figure 4.2 for Pika II data).

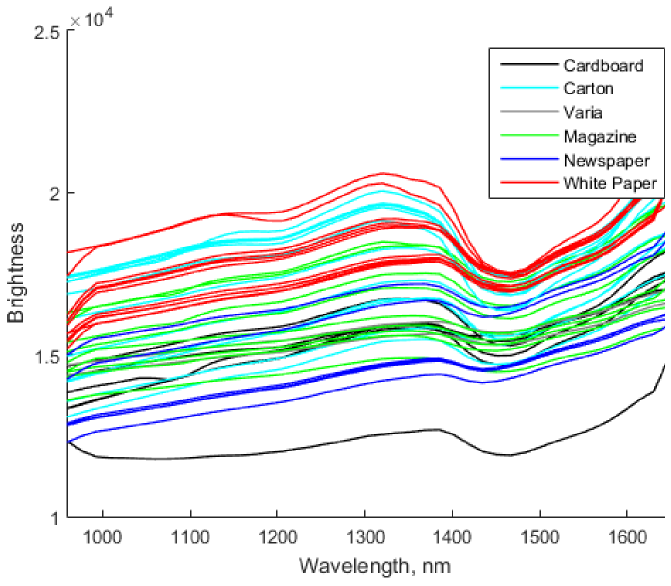


Figure 4.1 NIR spectra

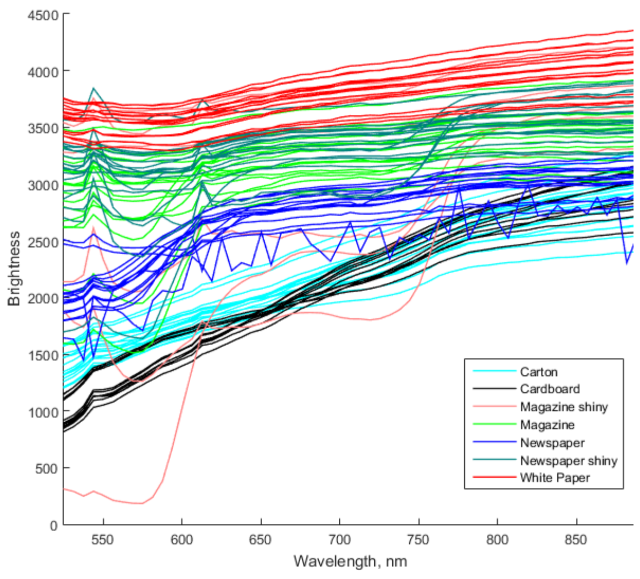


Figure 4.2 Pika II spectra

As can be seen, the spectra are quite similar among classes for NIR, and the same applies for Pika II as well. For NIR data, the most obvious variance between classes seemed to be at wavelength ranges 1400-1550 nm as well as right at the end of the range, at about 1600 nm. The spectral fingerprint of varying classes seemed to be similar in shape but expectedly varied in magnitude. Unlike Pika II data and visible light, data from the NIR could not be interpreted as color and the changes in the fingerprint must have been due to the chemical composition of the materials. Nevertheless, when examining the derivatives and filtered data of the NIR camera, it could be seen that it contained a lot of useful information in many wavelength ranges that needed to be investigated further.

The same did not apply for the Pika II data however. As mentioned before, the data was very noisy. What is more, the spectra of different classes seemed very similar, mostly flat throughout the whole range. Still, there were a few interesting factors that could be read from the spectral plots. Firstly, there seemed to be two sharp peaks at wavelengths 544 nm and 613 nm for a certain portion of the classes. These spikes can also be seen on Figure 4.2. These spikes, were very small in magnitude or not present at all for matte papers, but significant for glossy papers like magazine and newspaper covers. From that it could be concluded that the spikes were not noise or pixel error but turned out to be reflection from the illumination that was not filtered out by the polarizing filter. This fact could be used in feature selection, to select wavelengths where the variance between shiny materials and matte materials is the greatest.

As expected, color related results also showed on the spectral fingerprints of the Pika II. For example, *Äripäev* (an Estonian newspaper) had pink-toned paper and this was reflected in a step in intensity in the spectral fingerprint around the visible red wavelengths. Other cases like this could be seen for green and blue colors as well. However, these steps in intensity are clear indicators of color but they by themselves do not say much about the material and for information about the material the underlying trends of the spectral fingerprint had to be observed. The overall trend for all paper materials was continuously rising in intensity (except for the reflection spikes discussed above) and this was also consistent with the NIR data because the Pika II range ended around where the NIR range started. Another step in intensity could be observed for printed materials like newspaper and magazines right outside the visible range, especially for some dark print colors (large dark images).

4.1.2. Sample selection

The first step in analyzing the data was to group the obtained spectra together into classes based on Table 2.6 in order to form the classification calibration dataset. As suggested in previous research, k-means clustering was the simplest option to separate the background from the samples. A description of applying the k-means algorithm on the data and the various combinations with manual selection are described below.

In this thesis the algorithm in Pattern Recognition and Machine Learning Toolbox [22] with slight modifications is used. It is an optimized version of the MatLab k-means clustering algorithm and it uses the same principles as described above. The goal is to separate the background from the samples. As an output, the k-means clustering algorithm will give a binary $x \times y$ mask for the $x \times y \times \lambda$ hyperspectral image. K-means clustering is a very simple algorithm to separate data points into k partitions or *clusters*. For each cluster, the mean value of the data points belonging to the cluster is calculated and each data point belongs to the cluster with the nearest mean (mean with the smallest Euclidean distance). The process is iterative and is repeated until convergence is reached, meaning the mean values (cluster centers) do not change anymore.

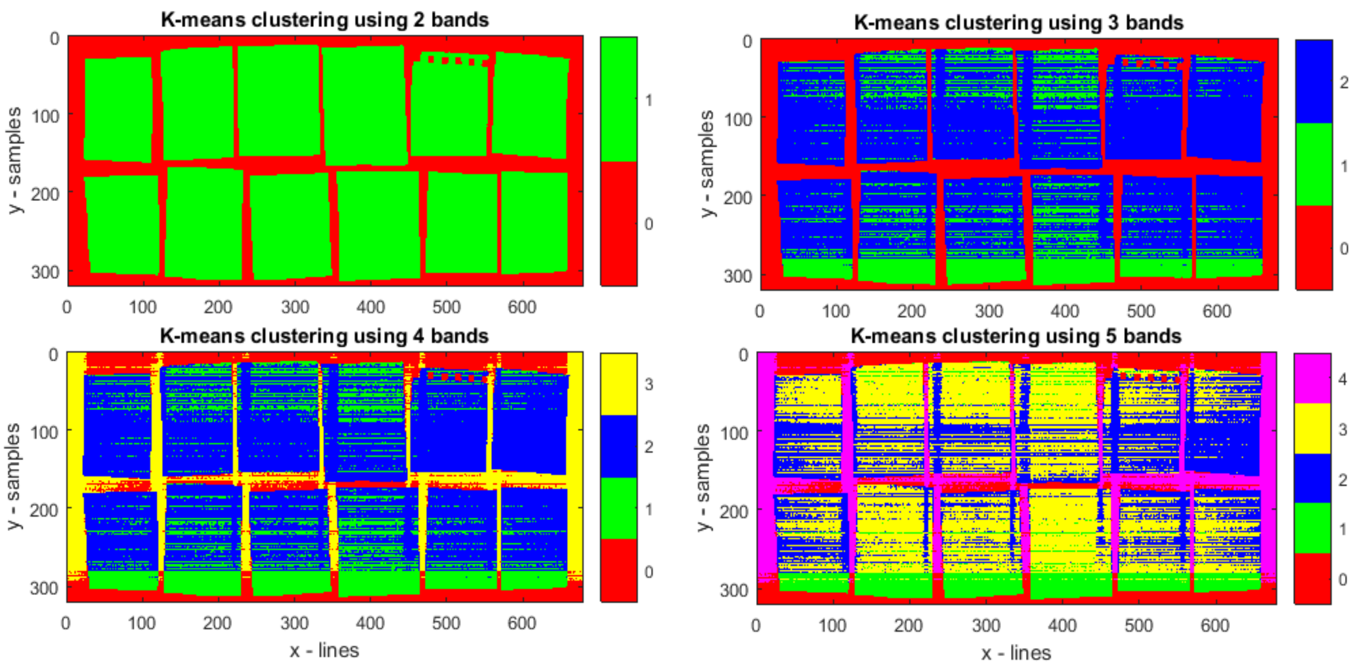


Figure 4.3 k-means clustering of clean white paper sheets with varying number of clusters, NIR, samples of white paper

The steps of the algorithm are the following:

1. Initializing the centers of the clusters. In [22] this is done by randomly labeling the data points.
2. Assignment of each data point to the cluster whose mean is closest in the Euclidean sense. The assigned labels are given in integers in [22].
3. Updating the means of the clusters to be the mean of the data points assigned in step 2.
4. Repeating steps 2 and 3 until convergence.

As suggested in [3], the k-means algorithm with two to five clusters should be tried. Figure 4.3 shows the results of the k-means algorithm with a varying number of clusters. As can be seen, using two and three clusters worked the best. Because using only two clusters is computationally the fastest and simplifies the algorithm (its output is already binary) then two clusters were used. However, k-means clustering works well on paper and cardboard samples with no print on them. Pixels of samples with dark print will be classified to the background class (see Figure 4.4) and therefore in these cases manual binary selection must be used.

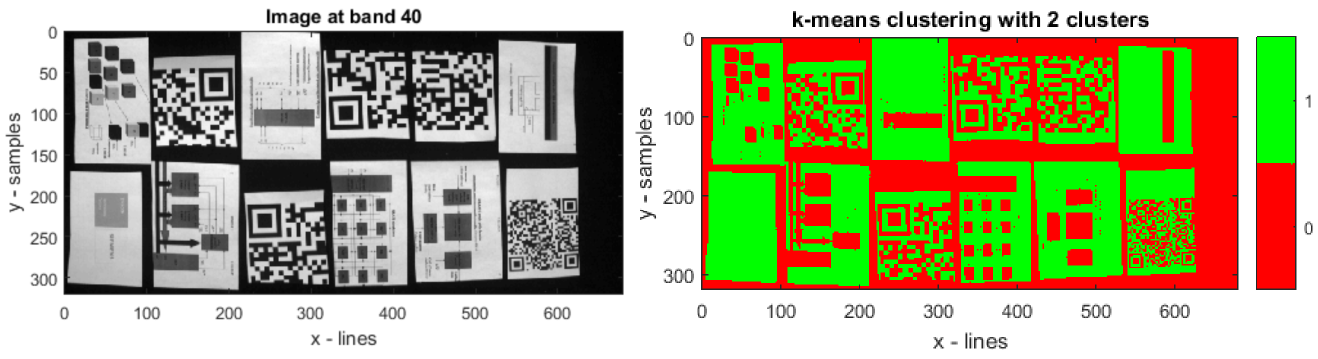


Figure 4.4 K-means clustering of paper with dark print. The dark print gets assigned to the background class. Manual selection must be applied here for classification.

As it turned out during the classification model calibration, separating the areas of the samples without any print was in fact desirable. The areas with print resulted in very large variances within classes and provided classification models with high error rates. Moreover, sample selection became crucial to the overall performance of the model, taking into account the accuracy as well as the speed of the algorithm. For example, the hyperspectral images of the NIR camera consisted of 44 wavelength bands in 320 samples per line and about 650-680 lines to cover the field of view, so the images were 320 × 650 × 44 of 14-bit depth. This meant that the images were quite large and it was not possible (and also redundant) to use all spectrums from all pixels to calibrate the method. Single pixels also contained a lot of noise and errors, showing as spikes in the raw spectral data (see Figure 4.2). To offset the large amount of data while acquiring various samples of different classes at the same time, different methods of sample selection were tried:

1. k-means clustering to select only the areas of the samples and then taking the median of each wavelength band of each class. This resulted as one spectral fingerprint per image. This method turned out to be suboptimal because it resulted in a quite small number of spectral fingerprints of different classes. Another disadvantage was the fact that, to get the spectral fingerprints, a large number of sample pixels were used which were obtained from areas of different illumination, camera angle etc. so the median spectral fingerprint included all of the samples but didn't really represent many of the pixels. See Figure 4.6.
2. Manual binary selection to select each paper sample separately and then taking the median of the pixels in that region of interest (ROI). This improved the sampling by resulting in greater number of samples from smaller areas of variance but by selecting the whole sample, areas with dark print were also included in the spectral fingerprint, lowering the integral value (area under the curve, meaning the overall intensity) of the spectral fingerprint significantly as well as altering some parts of the fingerprint due to different reflectivity of the ink. This could be avoided by manually selecting only areas of the sample without ink. This method had the same downside as the one above, namely the fact that the spectral fingerprint that was obtained did not represent many areas of the sample well due to the taking of the median.
3. Manual binary selection of the light/no print areas of the samples and taking *n*-uniformly distributed raw samples from the area. This method allowed the samples to be classified and a large number of samples to be taken without becoming uncontrollably large as well as having a good representation across all the samples due to no averaging taking place. However, this method did not smooth the spectral fingerprints in any way, resulting in samples of raw data. This meant that the samples included sensor noise and errors that had to be corrected later (see Figure 4.5). To get good representations of the spectrums within classes, the data was manually cleaned from false positives. Some of the samples contained erroneous data from the sensors that couldn't be fixed by filters nor manually so

the samples had to be removed from the training set. The downside is that the samples are taking from arbitrary locations on the sample and if the sample contains printed areas then the sample could include erroneous data.

4. Running a small median filter over the hypercube, taking the median along the spectral dimension and then repeating steps in 3. This made the obtained spectra less prone to contain sensor errors see Chapter 4.1.3 below. This method has the same downside as 3, namely that it can include spectra of printed areas.
5. The same method as 4, but adding a step of manual thresholding so that no samples would be taken form printed areas.

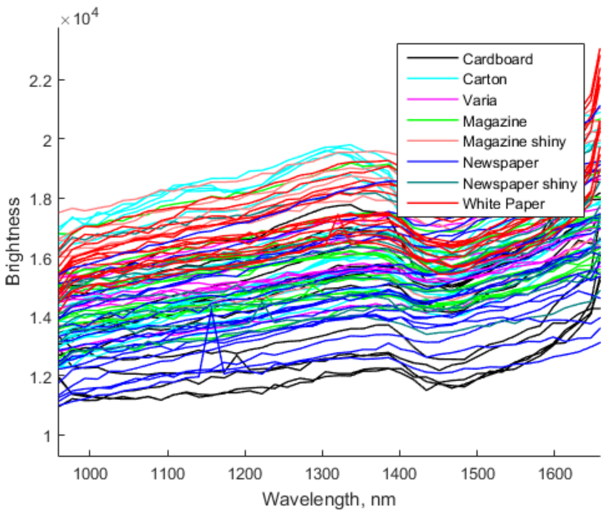


Figure 4.5 Raw spectras. Significant spikes and noise can be seen.

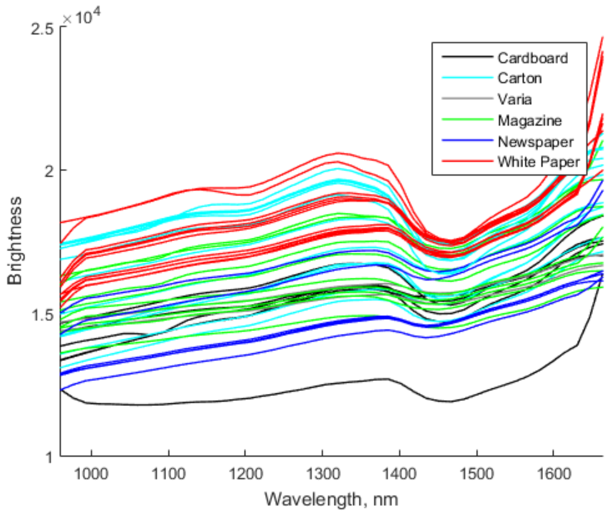


Figure 4.6 Spectra obtained by k-means clustering and taking a median

In addition to selecting samples, reading and parsing the hypercubes needed to be done. The description of scripts involved in the reading, parsing and plotting processes, as well as all the selection methods are in Appendix 7.

4.1.3. Data transformation

As seen on Figure 4.5, sensor errors result in spikes in the data. There were a few options to smooth the data to reduce the influence of noise. Firstly, as suggested in previous research, the Savitzky-Golay filter was used only along the third dimension of the hypercube, meaning only along the wavelength range of the pixel. For this filter, three parameters could be adjusted: window size, polynomial order and derivative order. For just filtering the derivative order was kept at zero but the same filter was later used for deriving the data as well. The impact of the filter can be seen on Figure 4.7, where one data point from Figure 4.5 was filtered. The blue plot represents the raw data, the orange plot shows a fairly good result using a window size of 7 and polynomial order 3, while the yellow plot shows a poorly chosen Savitzky-Golay filter with a too large window size, filtering out some of the relevant data. However, it can be seen that the high spike on the spectra (this spike can also be seen on Figure 4.5 as it is the same newspaper spectra) caused by sensor noise was not filtered out with Savitzky golay filter. On the contrary, the filter caused the spike to widen (due to the polynomial fit) and the edges of the spike to become sharp. To filter out this spike with Savitzky-Golay filter, a very large window size would have been needed and, as stated above, this would have eliminated useful information as well.

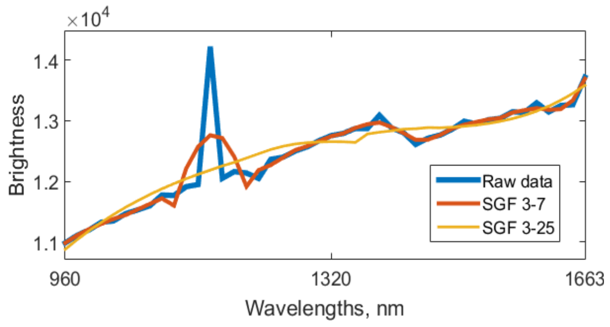


Figure 4.7 Effects of Savitzky-Golay filtering on one point of data

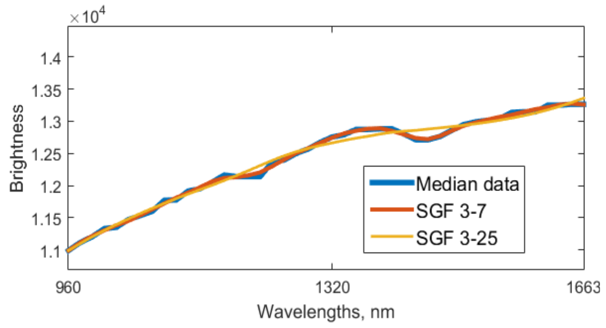


Figure 4.8 Effects of SGF after using median filter on the hypercube

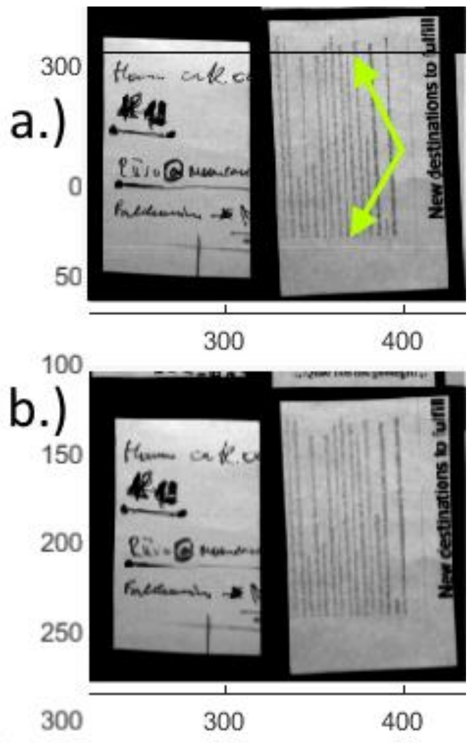


Figure 4.9 Effects of median filter on data at wavelength band 16 of the NIR camera. a.) before filtering and b.) after filtering

To deal with sensor error like above, a widowing median filter was used before the Savitzky-Golay filter. Figure 4.9 shows the sensor error effects on one slice of the hypercube at band 16 of the NIR camera, where errors were often observed. Since the hypercube is a 3D matrix then it was possible to control the dimensions along which the filter operated as well as the size of the window. A window size of just 3 bands along the third (spectral) dimension worked well in this case, as in Figure 4.9 where it can also be seen that no spatial information is lost due to the fact that the filter did not operate along the x- and y-dimensions. Figure 4.8 shows the same filters as applied in Figure 4.7 but applied after the median filter. The spike caused by noise is removed. It is important to note that for the NIR data this spike removal was desired but for the Pika II, as discussed above there were spikes in the measurements that were relevant and therefore a median filter was not used.

After the samples were selected and filtered, a number of standard pre-processing steps for PCA analysis were carried out. To reduce unwanted overall variation of the data within the classes, the data was normalized using the SNV method that was successfully used in previous work as well. The normalization removed the in class variation that was caused by the uneven lighting and other environmental influences. SNV (see Figure 4.10) method worked very similarly to the area normalization, MSC and PQN methods, but had slightly better results. Generally, the SNV normalization is used for scatter correction which is a common drawback of NIR spectroscopy but the same filter was used with good results on the Pika II data as well.

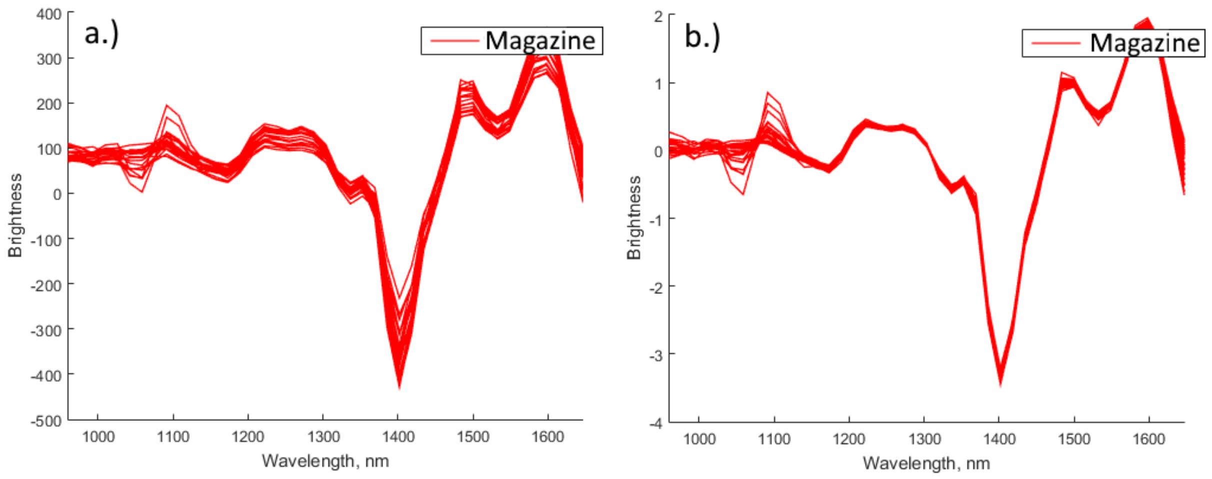


Figure 4.10 Magazine spectra measured with NIR camera. The samples are medians of selected areas and the first derivative of the spectral fingerprint a.) derivative spectra; b.) derivative spectra with applied SNV.

As can be seen of Figure 4.10, the SNV changed the data in two ways. First, it lowered the variance of the data, shown by the plots converging more closely together on Figure 4.10 b.). Secondly, it moved the data closer to the zero point. This is a feature of the SNV method [20] and the transformation could be named equal to a total-energy scaling (normalization) applied to a centered signal.

For principal components analysis and partial least-squares methods it is generally advised to use centering of the data to align the data around the origin of the n-dimensional space. The most common centering method is mean centering, which is subtracting the mean of all of the values within one variable (in this case, one wavelength band) from the corresponding variable. This results in the data clustering around the origin and this also emphasizes the differences from the overall mean of the data. However, since the centered data will depend on the overall mean then it is important that the samples are represented equally among the classes. Another common and often suggested and required preprocessing step is to scale the data so that variables of different units can be taken into account equally. In analyzing spectral data however the data is already in the same units of light intensity. Even then it is generally advised to perform scaling so that the variance of the spectra would be of one unit because otherwise the higher peaks in the data become more relevant than smaller variations [21].

By trying out various combinations of all the steps in Chapter 4.2, the pre-processing steps were chosen. The quality of the pre-processing was evaluated in cooperation with the classification and cross-validation methods discussed below. The final selected combination of pre-processing for respective data is shown in Table 4.1. Some of the pre-processing steps that had good results on spectra of separate classes, were not used in the pre-processing of the training and prediction because they caused unnecessary errors (such as SNV).

Table 4.1 Pre-processing steps

Step	**NIR**	**Pika II**
Sample selection	12 evenly distributed data points per sample, thresholding to remove dark areas, median filter	Median of each sample that was previously thresholded to remove dark areas
Pre-processing	Mean centering, 2nd derivative with window size 11, polynomial order 2.	Autoscale

4.2. Principal Components Analysis

Principal Components Analysis (PCA) is a very common technique while working with big data and in interpreting influence of various variables on the data. PCA is not a classification method but it gives plenty of information about the dataset, it can also be used for variable selection. PCA analysis was done on both datasets, the discussion and results are given below. The following discussion is based on the analysis on the data acquired with the NIR camera, but the same applied for the Pika II datasets as well (the plots for the same Pika PCA analysis are in Appendix 5). For the PCA analysis MatLab toolbox *PLS_Toolbox* [23] was used.

Simply put, PCA finds a linear subspace in the λ-dimensional space (λ is the number of wavelength bands) with orthogonal axes (principal components, PCs) where the variability of the data is the largest. The principal components are linear transformations of the original data points. The goal of PCA is to represent the same data within a subspace of minimal dimensions. Along with the PCA model, the PLS_Toolbox runs cross-validation, iteratively separating the training set into sub-sections with which to validate the model. The method of cross-validation in the tests carried out here was Venetian blinds cross-validation with 10 iterations and one sample per "blind" or selection to the cross-validation. Cross-validation helped to estimate how well the model fit the data and how data points outside the training set would fit the data as well. What is more, cross-validation helped in choosing the number of latent variables (number of PCs) of the PCA, which was a crucial step in selecting a good model. The PLS_Toolbox suggests a number of PCs based on the root mean square error of calibration (RMSEC) and the root mean square error of cross validation (RMSECV). Figure 4.11 shows how the number of latent variables affects the RMSEC and RMSECV. In general, the number of principal components should be chosen so that by adding one additional principal component does not improve the RMSEC and RMSECV significantly. For example, for the model in Figure 4.11, the optimal number of principal components is four, because having five PCs would not have a significant improvement of the RMS errors anymore.

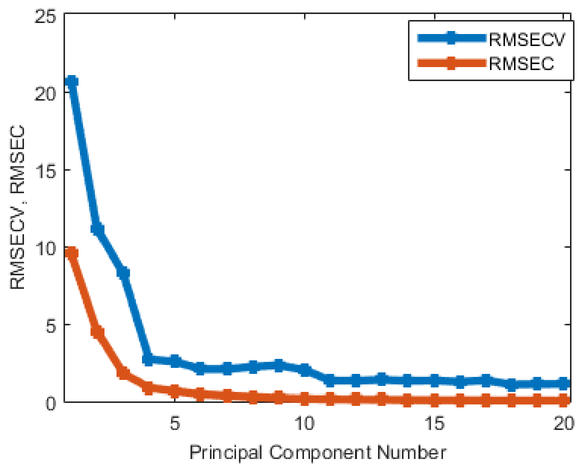


Figure 4.11 Number of PCs vs the RMSEC and RMSECV

Table 4.2 Variances and RMS errors in the PCA model

#	% var. in PC	% cum. var.	RMSEC	RMSE CV
1	81.01	81.01	9.52	20.63
2	14.81	95.82	4.466	11.1
3	3.50	99.32	1.804	8.245
4	0.53	99.84	0.8644	2.724
5	0.07	99.91	0.6504	2.554
6	0.05	99.96	0.4526	2.082
7	0.02	99.97	0.3515	2.077
8	0.01	99.98	0.2762	2.234

Table 4.2 shows how the variance of the data points is captured in the PCs. Since PCA finds PCs where the variance of the data points is the largest then the first PCs also capture the most of the variance. So in Table 4.2 the first principal component captures 81% of the total variance. Four PCs were chosen with the reasoning mentioned above, so the total variance captured by the four PCs is 99,84%. The next PCs held very little extra information. The RMSECV was expectedly higher from the RMSEC due to the fact that the cross-validation data points were not used in the calibration of the model and the PCA models are quite prone to over fitting (the more PCs, the more likely the over fitting). When the number of PCs is chosen, the fit of the model can be further investigated by the Q-residuals and Hotelling T^2 values. The Q-residuals showed the percentage of how much of the data point *cannot* be explained by the model. Therefore a sample with a large Q-residual has unusual variation outside the model. For example, on Figure 4.12 there are a few data points belonging to the Magazine, Magazine Shiny and Newspaper data points that have quite large Q-residuals, meaning that these data points do not fit the model very well and they should be investigated further to see whether they are erroneous samples. The more PCs are used in the calibration of the model, the better the Q residuals calculation, with the downside of over fitting when using a large number of PCs. If nr of measured variables is equal to the number of PCs then calculation of Q-residuals is not possible. That means outliers of the model cannot be detected. This was not the case in Figures 4.12 to 4.18 because all of the variables (wavelength bands) were used, but had to be taken into account when selecting variables. The Hotelling T^2 showed how much variance the data points have *inside* the model. A large T^2 value for the Cardboard and Carton classes indicates unusual variation inside the model.

When looking at the scores and loadings plots, what could be read out from the PCA plots was the fact that using data points taken from dark or printed areas of the materials resulted in very poor separation of the classes. Figure 4.13 shows the scatter plot of scores of principal components 1 and 3 of training data where dark and printed areas were not filtered out. Figure 4.16 shows the same scatter plot of only data points of light and/or uncoated areas of the materials. Sample selection and pre-processing are the same for the datasets, as well as the number of latent variables used; the only difference being that for the second dataset a simple manual thresholding was used, so that no spectra from areas covered with print would be selected for the training set.

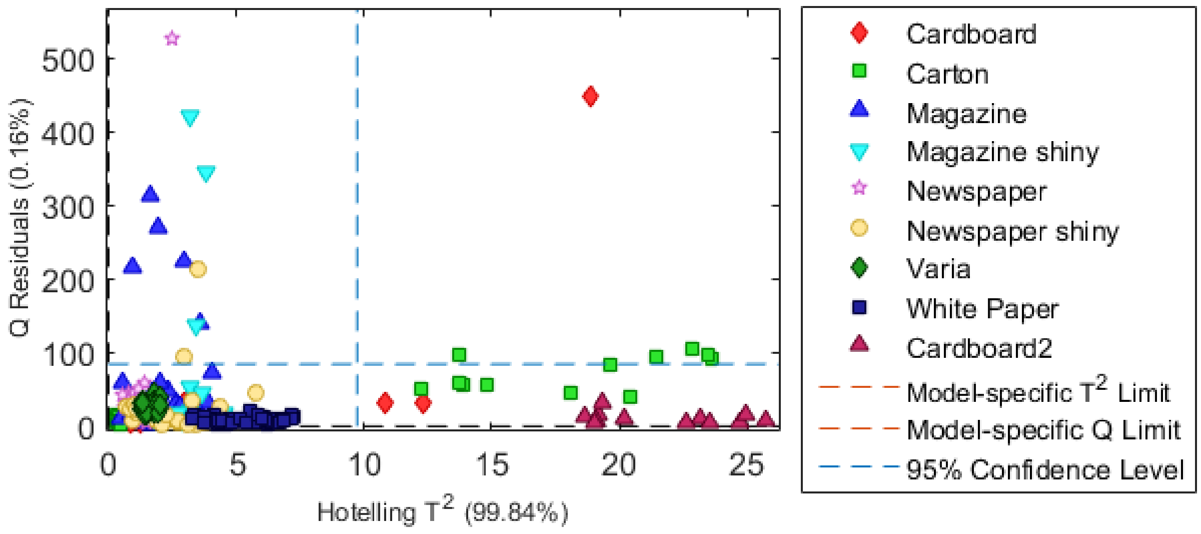


Figure 4.12 Q-residuals and Hotelling T^2 plots, showing how the data points fit the model. In parentheses the percentage of the variance covered.

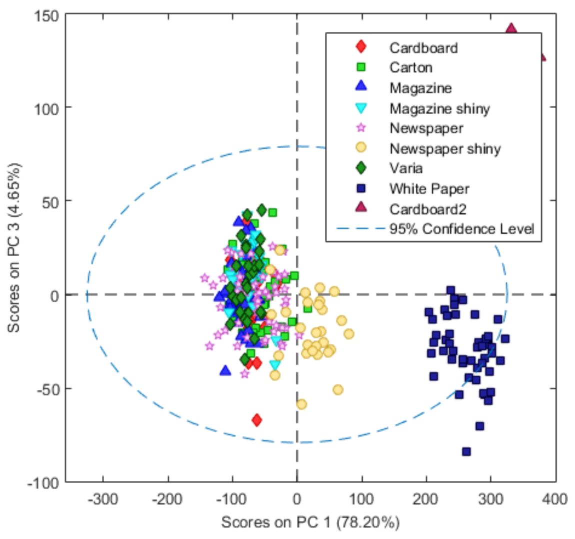


Figure 4.13 PCA on samples containing print. Scatter plot of scores on Principal Components 1 and 3. Plot generated with PLS_Toolbox

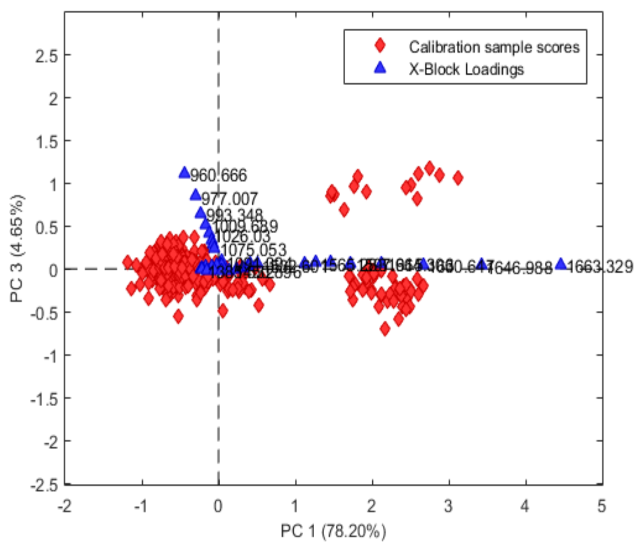


Figure 4.14 PCA on samples containing print. Biplot of scores and loadings of Principal Components 1 and 3. Plot generated with PLS_Toolbox

On Figure 4.13 it can be seen that only the spectra of white paper (dark blue squares) was separated from the main cluster of data points (in fact, behind the legend there was another cluster of cardboard class). All of the other classes were scattered around the origin of the principal components. It could also be seen that the data points were scattered with quite large variance within classes as well as overlap of the data points projected onto the principal components' axes. Similar behavior of data points was observed on scatter plots of scores of other principal components. On the other hand, on Figure 4.15 it can be seen how the variation within a class was reduced significantly for most of the classes. What is more, some of the classes, for example ‘Newspaper shiny’, have moved away from the other classes. It is worth mentioning that cluster separation does not necessarily mean that the classes would be separable from each other. However, here the classes could be declared separable if they are visually separable on the scores plot *and* they fall within the 95% confidence limit marked with a blue dotted line on Figures 4.13 and 4.15. So on Figure 4.15 the *Cardboard2* cluster fell out of the confidence limits, meaning that the model did not fully explain the separability of this cluster. Nevertheless, comparing Figures 4.13 and 4.15, it could be said that it was more likely that classification using only areas not covered with print would be more accurate. The print would either have to be classified separately (which would include the risk of being misclassified as background) or using some further methods of machine vision to include the black print to the appropriate class.

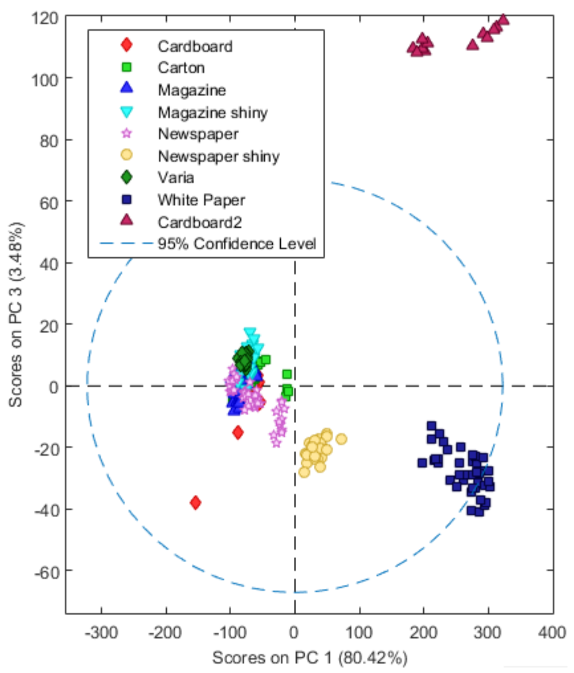


Figure 4.15 PCA on samples *not* containing print. Scatter plot of scores on Principal Components 1 and 3. Plot generated with PLS_Toolbox

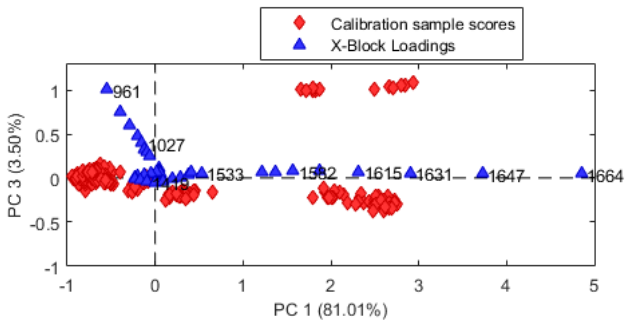


Figure 4.16 PCA on samples not containing print. Biplot of scores and loadings of Principal Components 1 and 3. Numbers above loadings markers are the wavelength bands (some omitted for clarity). Plot generated with PLS_Toolbox

Figures 4.14 and 4.16 show the loadings and scores biplots for the respective datasets within PC 1 and 3. Scores and loadings are correlated in the sense that when a variable (in this case, one wavelength band marked as blue triangles on Figures 4.14 and 4.16) has a large value on the loadings plot then this variable has directly influenced the data points (marked as red diamonds) that also have a large value in that principal component. For example, of Figure 4.16 it can be seen that a cluster of data points was separated from the others in the first sector of the plot (PC1 > 0; PC2 < 0). From Figure 4.15 it can be determined that this cluster is of the white paper class. Figure 4.16 also shows a number of variables (wavelength bands) that have an increasingly large positive value in PC1. The meaning of this was that wavelengths 1549 – 1663 nm were directly responsible for the separation of the White Paper cluster as well as the Cardboard2 cluster, but for the latter wavelengths around 961 nm also played an important role. Analyzing the loadings and scores plots in this manner gave an overview of what wavelength bands play a role in the class separation. The same information about all of the four principal components used in Figures 4.15 and 4.16 is summarized by the variables versus loadings plots on Figure 4.17. The sum of the squares of the loadings for each principal component is 1 and the loadings essentially show the contribution of the variable to the principal component.

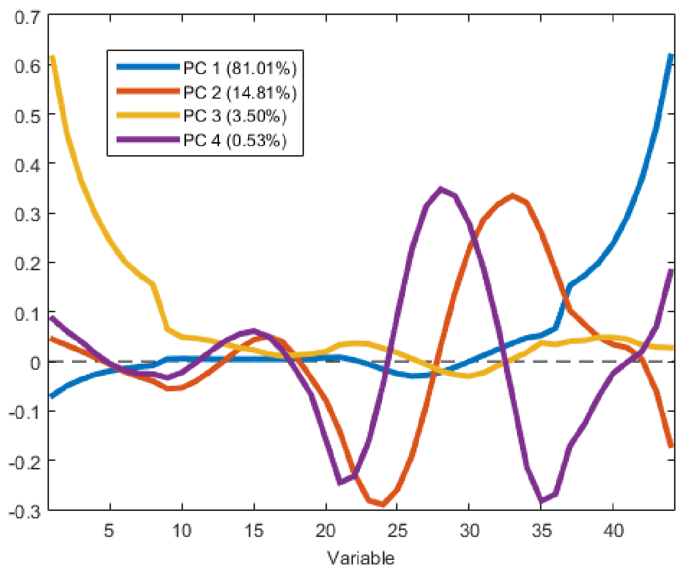


Figure 4.17 Variable (band number) vs. loadings for PC1 – PC4.
Plot generated with PLS_Toolbox.

So for example, the previously mentioned cluster of white paper data points had a high score on the first principal component and it could be seen that the variables from the end of the wavelength band range were responsible for this. This can be confirmed on the loadings plot as the loadings for the first component are relevant at the range 1549 – 1663 nm (on Figure 4.17 the variables are labeled by their index rather than the actual value, so 1549 nm =

variable 37 and 1663 nm = variable 44). In each principal component, the variables with the highest peaks in the loadings plot are the most relevant to this component (the sign of the loading value is not relevant). Table 4.3 summarizes these most relevant variables for each principal component (see full table in Appendix 4). This table can be used for variable selection, where only one band from the relevant range can be selected, because the other values in the same range are highly correlated with the selected variable. So for instance from the wavelength range 1549 – 1663 nm only the last band at 1663 nm can be selected because its contribution to the principal component is the greatest and the others with high loadings in this principal component have the same effect on the data (all the before mentioned PCA data could also be accessed via the PLS_Toolbox model, not only the plots).

Table 4.3 Variables with largest contributions to the loadings of principal components, NIR

	PC1	**% of PC**	**PC2**	**% of PC**	**PC3**	**% of PC**	**PC4**	**% of PC**
Ranges	1664 nm	38,0	1468-1500 nm	31,6	961-994 nm	72,0	1386-1418 nm	33,2
	1615-1647 nm	44,5	1321-1353 nm	23,0			1500-1533 nm	19,8
			1663 nm	2,9				

4.2.1. Summary of results of PCA for the Pika II spectra

The PCA analysis was carried out on the Pika II data as well. The Pika II data contained less variance between classes. Therefore the classes were not divided into the actual paper materials, but into three sub-groups: white paper, cardboard (and carton) and glossy papers (glossy newspaper and magazine). The figures in Appendix 5 show the scores plot and the loadings plot of the PC1 and PC3, as in these principal components the class separation was the most obvious. The white paper and cardboard subgroups had fairly small variance within the class, as seen on A.1. The magazine subgroup displayed the largest variance. From the loadings plot A.2 it can be seen which wavelength bands are most relevant for each principal component. The loadings plot confirms the previous discussion of the fact that the spikes caused by the illumination are relevant markers in separating glossy materials from matte materials. The most relevant wavelengths have been added to Table 4.4. The PCA analysis of Pika II spectra showed that it was possible to separate the spectra into a few subgroups of materials. However, the NIR data provided in general the same kind of separation possibilities, with more data to refine the classification so materials could be separated. A disturbing factor for the Pika II spectra was the amount of noise present in the data. The Pika II wavelength range tests should be repeated using more illumination to see if that gives better results. The only information that the Pika II provided and the NIR did not was visible color information. But for color detection a hyperspectral camera is highly over dimensioned, a simple RGB camera would resolve the colors as successfully as the Pika II. The Pika II data acquisition should be revised but the data acquired in this work was not used for further classification, only NIR data was used in the following chapters.

As mentioned above, if the classes are well separated within the first PCs on the scores plots then the loadings with large values in the loadings have good potential in being relevant in the separation of classes and can be selected as contributing variables. Out of the 44 wavelength bands for NIR data, six variables were selected. Out of the 59 bands of Pika II data, also six variables were selected (Table 4.4).

Table 4.4 Summary of selected wavelength bands

Selected wavelengths						
NIR	961 nm	1337 nm	1402 nm	1484 nm	1517 nm	1664 nm
Pika II	537 nm	544 nm	550 nm	575 nm	738 nm	888 nm

4.3. Classification

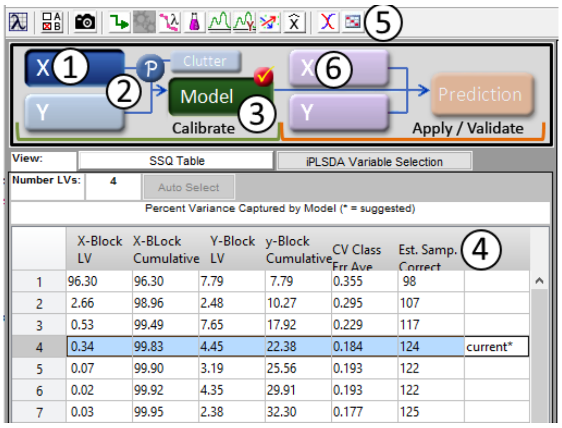
The PCA was performed on the data to get some insight about it but as mentioned before, it is not a method of classification. To discriminate classes, three methods were tried: PLS-DA, SVM and k-nearest neighbor (k-NN) methods, all of which were also mentioned with good results in previous research. The classification was run on the full set of data, not with the variables selected with the PCA method to see how accurate the model could get. In the Chapter 4.3.1, an overview is given of how the model calibration was done using the PLS_Toolbox. Next, a description of how the model quality was assessed is given. Then the results are reviewed, starting with SVM and k-nearest neighbors because they were deemed less suitable than PLS-DA and therefore there is less information concluded from these methods. After the SVM and k-nearest neighbor analysis a detailed overview of the PLS-DA application was given and the results of the classification were presented in Chapter 5.

4.3.1. Calibrating the classification model

The model calibration, analysis, cross-validation and initial validation were done using the PLS_Toolbox GUI [22]. This made it simple to review the results and recalibrate accordingly. The following overview is given so that the results of this work could be reproduced when needed.

Firstly, the data from the cameras needed to be prepared for the toolbox. That meant selecting, parsing, reading and reshaping the data to be suitable for the toolbox. Also, since this process is a calibration where the data needed to be previously labeled then the methods described above for sample selection were used (see Table 4.5 for the chosen classes). The reading and labeling process was done using scripts described in Appendix 7. The following steps were:

1. Importing the data to the PLS_Toolbox model X-block (Figure 4.18 1.). The X-block contained the data in a $M \times N$ matrix, where M – number of samples (single pixels from the hypercube) and N – number of variables (wavelength bands).



View: SSQ Table | iPLSDA Variable Selection

Number LVs: 4 Auto Select

Percent Variance Captured by Model (* = suggested)

	X-Block LV	X-BLock Cumulative	Y-Block LV	y-Block Cumulative	CV Class Err Ave	Est. Samp. Correct	
1	96.30	96.30	7.79	7.79	0.355	98	
2	2.66	98.96	2.48	10.27	0.295	107	
3	0.53	99.49	7.65	17.92	0.229	117	
4	0.34	99.83	4.45	22.38	0.184	124	current*
5	0.07	99.90	3.19	25.56	0.193	122	
6	0.02	99.92	4.35	29.91	0.193	122	
7	0.03	99.95	2.38	32.30	0.177	125	

Figure 4.18 The PLS_Toolbox GUI [22]. 1. Load and edit the X-block data. 2. Pre-processing. 3. Calibrate model. 4. Review the calibration data and fit. 5. Select plots for more information on the calibration. 6. Load validation data.

2. Importing the labels of the data. The labels were in a vector of $M \times 1$, containing strings or the class names (for example, “Cardboard” or “White paper”). There was two methods of how to import the labels and connect them to the data:
   a. Loading the labels into the Y-block of the model
   b. Editing the X-block data so that the row labels tab included the labels as classes.

In this work, the second option was used. This gave the benefit of seeing the classes on the plots as different symbols etc. as well as the option to combine classes.

3. Reviewing the X-block data to exclude outliers from the model.
4. Selecting the preprocessing (see reasoning above in Chapter 4.1.3), (Figure 4.18 2.)
5. Loading the model. Reviewing the results (Figure 4.18 3., 4. and 5.)
6. Assessing the quality of fit of the model by the cross-validation (Figure 4.18 4.)
7. Loading validation data into X-block-validation and predicting, assessing the results (Figure 4.18 6.)

The toolbox included the cross-validation option. Cross-validation was the main mteric in assessing the fit of the model to the data. In the calibration done in this work, venetian blinds cross-validation was used with 10 data splits and one sample per blind. In this way, for each iteration of the calibration, 10% of the dataset was left out of the calibration and was used for validation of the model. This produced an average cross-validation (CV) error for each model and, if the modeling groups were chosen, for each group. Obviously, the lower the cross-validation error the more likely that the model would work on new data as well so this was a very valuable insight to how to model was fitting the data.

Table 4.5 Classes used in model calibration

Label number	Classes
1	White paper (WP)
2	Magazine (MG)
3	Newspaper (NP)
4	Carton (CA)
5	Cardboard (CB)
6	WP dark print (WP D)
0	No class – N/A

Table 4.6 Average cross-validation errors when classifying all classes with one model was attempted

	PLS-DA	SVM	k-NN
Average CV-error	21,7%	8,7%	7,7%
Pre-proc.	SGF, 2nd deriv. (15-2) mean center	SGF, 1st deriv. (15-2)	autoscale, 1st deriv. (15-2)

To assess how the discrimination methods would work on the data, the cross-validation error of all of the models was compared. The pre-processing of the data was not the same for all of the methods, but chosen so that the CV error would be minimal (see Pre-processing in Table 4.5). Different pre-processing steps were taken because the methods generate the discriminators differently and need different kinds of data for input, the biggest difference being that the SVN and k-NN methods do not require mean centering but for the PLS-DA it is highly recommended. As it turned out (see Table 4.4), the paper materials' spectral fingerprints were too closely related to correctly and reliably separate into classes with just one classification model. Many classification models had to be used together, firstly to separate materials into groups and then use other models to separate the sub-groups further. This was easily achieved with the PLS_Toolbox's Hierarchical Model Builder.

4.3.2. SVM and k-nearest neighbor discriminant analysis

Support Vector Machine classification is a linear supervised classification process. The cross-validation errors of each class of the model are shown in Table 4.7. See full data in file *SVM_model_allClasses.txt*.

Table 4.7 Cross-validation errors of classification with the SVM model, within classes

	White paper	Magazine	Newspaper	Carton	Cardboard
c-SMV CV-error	3,3%	6,0%	5,3%	12,5%	8,6%
Nu-SVM CV-error	2,0%	15,2%	12,6%	17,2%	12,6%

Table 4.8 Cross-validation errors of classification with the k-NN model, within classes

	White paper	Magazine	Newspaper	Carton	Cardboard
k-NN CV-error	13,9%	7,3%	6,6%	15,9%	10,6%

A downside of the CVM classification was that compared to the PLS-DA method, it was very slow. As the method remaps all of the data points then a lot of calculations on the data were carried out. A faster version of SVN was the nu-SVM classification, but it was far less accurate (see *nu-SVM_model_allClasses.txt*).

The k-nearest neighbor (k-NN) algorithm is one of the simplest machine learning or classification algorithms. It does classification only locally, basically counting the classes of the data points' nearest neighbors in the Euclidean sense and classifying the data point to the class that would have the largest amount of neighbors in the same class. The within class CV-errors are shown in Table 4.8 (all details in *k-NN_allClasses.txt*). The k-NN method was fast and quite accurate so it was considered in using in some of the modeling steps of the total hierarchical model. The problem with the k-NN method was that despite being quite fast in the calibration process (where the number of data points is quite small), it was extremely slow in the prediction process because the hypercubes contained about 320 times 680 spectra, each of which needed to be compared to every data point in the training set. Another downside of using the k-NN method in the PLS_Toolbox was that class grouping was not possible so if the goal was to group data points into groups of classes and not specific one-material classes (classes 1-5) then the labeling needed to be done manually. Both of these methods had their benefits but the PLS-DA seemed more fitting and faster so the further classification was done using the PLS-DA.

4.3.3. Classification using the PLS-DA method

The PLS-DA method was chosen because it was much faster to run than the SVM and k-NN methods. Additionally, the PLS-DA method in the PLS_Toolbox allowed the classes to be easily grouped to make hierarchical modeling simple. Furthermore, the PLS-DA method generated far more information about the model than the other methods. The information was very similar to the principal components analysis, outputting scores and loadings plots as well as prediction information. Table 4.9 shows the cross-validation errors of classes when all classes are classified within one model (see all details in *PLS-DA_model_info.txt*). As can be seen, the errors are much higher than in Tables 4.7 and 4.8. Nevertheless, the results of separating the classes first into subgroups and then into classes was more successful.

Table 4.9 Cross-validation errors of classification with the PLS-DA model, within classes

Class	PLS-DA CV-error
White paper	20,7%
Magazine	19,4%
Newspaper	18,3%
Carton	18,6%
Cardboard	31,7%

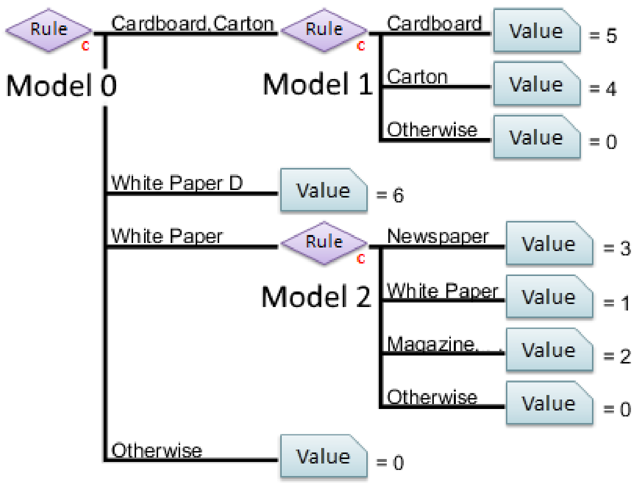


Figure 4.19 Three-tier hierarchical classification model. Generated in the PLS_Toolbox

The three-model hierarchical classification model concept is shown on Figure 4.19. The first model (Model 0) was responsible for separating carton and cardboard from the other classes. Model 0 also separated a small group of dark print on white paper into a separate group, but classifying real data into this class was very rare. So the model discriminates between three classes: cardboard or carton, white paper (and MG and NP) and WP D. Model 1 was responsible for separating cardboard from carton and Model 2 discriminated between white paper, magazine and newspaper. If the data point did not fit any of these classes then it was classified as N/A or value 0. A description of each model's properties is given below and the results of classification are given in Chapter 5.

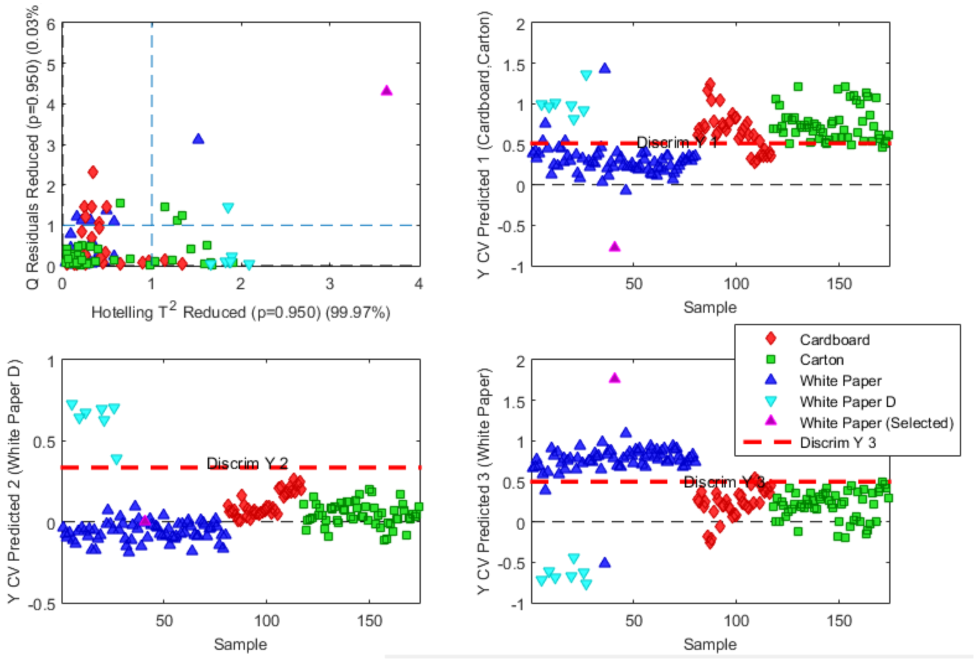


Figure 4.20 Model 0 separating cardboard and carton from other classes. Figure generated with PLS_Toolbox

For Model 0 four latent variables were used, in total explaining 99,97% of the variance of data. Figure 4.20 shows the fit of the model and the class separation. From the first plot, the Q-residuals vs. Hotelling T^2, it can be seen that most of the data points are within the 95% confidence limits (blue dotted line) of both axes, meaning that generally the data points fit the model and that they have reasonable variance within the model. There is one white paper data point highlighted with pink. This data point had high values for both the Q-residuals and the Hotelling T^2, meaning that it could have been an outlier. Excluding outliers may have helped to increase the quality of the model, so it had to be considered. However, this data point didn't seem to make the fit of the model worse because, as seen on the samples vs. prediction plots, it was always classified correctly with high probability values. The samples vs. prediction scatter-plots on the figure show how the training set of data is discriminated within the model. The red dotted line is the threshold for the model, which is calculated based on the Bayesian method (for more information on threshold calculation, see [23]). As seen on the graphs, the data points that have a larger probability value than the threshold of the class, are considered members of that class. The cross-validation errors by class are in Table 4.10. It is important to note here that only white paper data points (and not NP or MG) were used for the separation of the "Other" class. Nevertheless the model trained on white paper seemed also to apply to magazine and newspaper, as the validation showed (see Chapter 5).

Table 4.10 The cross-validation class error for the PLS-DA model separating cardboard and carton from the other classes

Class	Cardboard, Carton	WP D	Others
CV error	14,3%	7,1%	2,4%

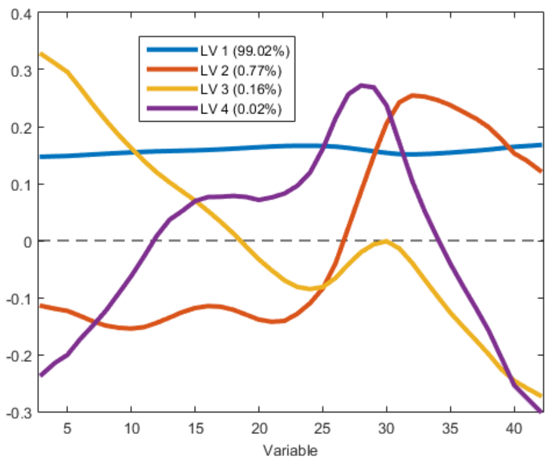


Figure 4.21 Loadings of Model 0. In brackets – the variance explained

The loadings values for the model were inspected to confirm the results of the PCA analysis that six wavelength bands could be used instead of 44 to classify the materials. Figure 4.21 shows the loadings plot of the four principal components used. The most noteworthy difference from Figure 4.17 is that the first PC that explains 99% of the variance, uses all of the variables equally. The other PCs that explain much less variance (a total of about 1%) seem to confirm the fact that the wavelength bands at the range limits and at bands around band 30 are important for the classification. Regardless of the other principal components, the first PC explains so much variance that dropping some of the wavelength bands from the dataset might have an extremely negative effect on the quality of the model. This was confirmed also by the selectivity ratio which showed essential variables for each of the classes. The trend in the selectivity ratio plots was the same: in general all the wavelength bands are required, but the most crucial ones are the bands around band 30 (around 1300 – 1500 nm). Therefore, in the following analysis, only full sets of data were used.

Model 1 separated cardboard from carton. The error percentage of this classification was quite high due to the fact that the spectra of the data were very similar and the differences could not be brought out by pre-processing. The best separation seemed to work with only autoscaling the data and not applying any more filters. The cross-validation error of this model was 26,2%. Once again four principal components were used. The Q-residuals vs. Hotelling T^2 and the discrimination plots looked very similar to the plots in Figure 4.20 meaning that the data fit the model quite well.

Model 2 discriminated between the WP, MG and NP classes. Figure 4.22 shows the fit of the model and the class separation. On the plots, the cardboard and carton data points are included to show how they would fit into the model but in practice carton and cardboard spectra should not reach this model as they would be filtered out by Model 0. The Q-residuals vs. Hotelling T^2 plot confirms the fact that this model is not very well suited to classify cardboard and carton data (and WP D), but suits well for the other classes. The similarities of newspaper and cardboard spectra can be seen on the newspaper class discrimination plot, where cardboard is largely classified as newspaper. This behavior was seen in running the model on real data as well and printed areas on newspaper material were classified to the cardboard class (see Chapter 5 below for more details).

Table 4.11 Model 2 cross-validation errors per class

Class	CV error
WP	14,0%
NP	5,3%
MG	16,7%

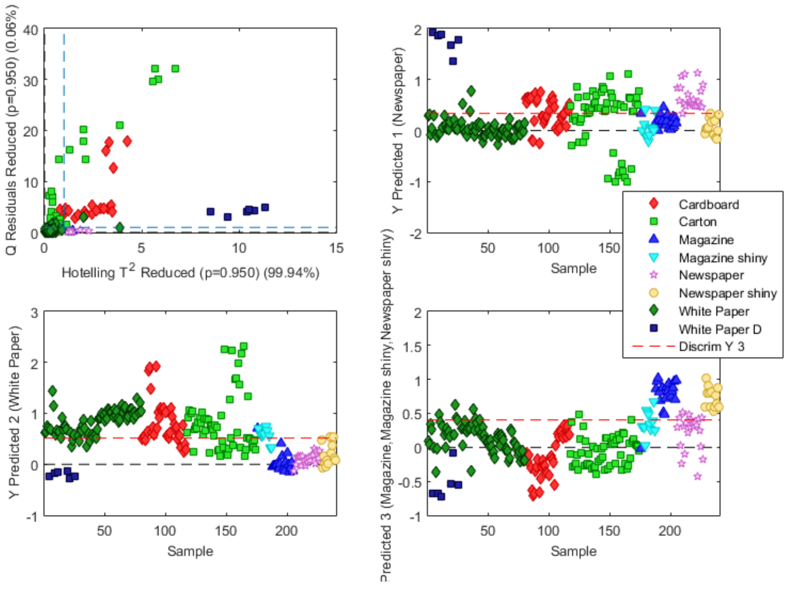


Figure 4.22 Model 2 separating WP, NP and MG. Figure generated with PLS_Toolbox

The magazine class contained data points from three materials: magazine sheets, magazine covers (magazine shiny) and newspaper covers (glossier material than newspaper, resembles magazine sheets in both spectra and physical qualities). They were put together into one group because these data points had very little difference in the spectra. The magazine cover however did not classify well into the magazine class and that was also seen later in the testing phase. As can be seen from Tables 4.10 and 4.11, the cross-validation error was still quite high. To counteract the high misclassification, a simple statistical generalization was performed on the classified data of the hierarchical model, described below in Chapter 4.3.4. The three models were combined into one hierarchical model as seen on Figure 4.19 and the results were output to MatLab to process further. An example of an output of the hierarchical model (reshaped to regain the spatial information) is in Figure 4.24.

4.3.4. Generalized classification method

The output of the hierarchical classification model was a $N \times 1$ matrix, where N was the number of data points in the hypercube, containing the label of the class. Using MatLab scripts (see Appendix 7), the hypercube was reshaped into an image of the same size as the hypercube's spatial dimensions. Figure 4.23 shows an image cropped from the hypercube at wavelength band 23 (1320 nm) and Figure 4.24 shows the reshaped output of hypercube *Validation 1*. Each color represents an integer label number (for example, 1 for white paper, see labeling in Table 4.4). As can be seen, each pixel is classified regardless of its surrounding pixels and therefore there are many classes present within one sample of paper. Also, the pixel-by-pixel classification error ratio is quite high. As can be seen, the areas of the samples containing print, especially dark print, are often classified differently as the areas of the sample that do not contain paint.

What is interesting as well is that the illumination of the scene also played a role in the classification. Figure 4.25 shows samples of white paper that contain some print. The bottom row of the samples shows exactly how much the illumination affected the classification. All the pixels (except the pixels containing print) should be classified into the white paper class (as they are in the top row) because the model worked very well on white paper. Nevertheless, in the bottom row the samples are misclassified and the shape of the misclassified areas showed that it was not random misclassification or material-based misclassification but misclassification caused by spatial environmental variables. Variations in illumination would explain these errors well. However, these kinds of errors do not show the poorness of the model but the fact that the camera setup should be adjusted. It is clear that in an industrial setting the samples would be linearly moving past the sensor and the illumination area would be one line where the illumination could be held constant instead of a rectangular region where the illumination very highly depends on the positioning of the lamps.

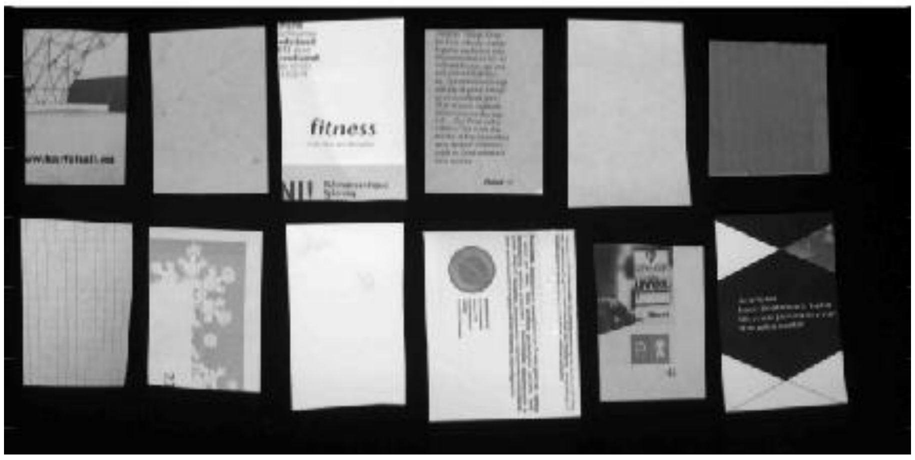


Figure 4.23 Image of *Validation 1* hypercube, cropped at wavelength 1320 nm

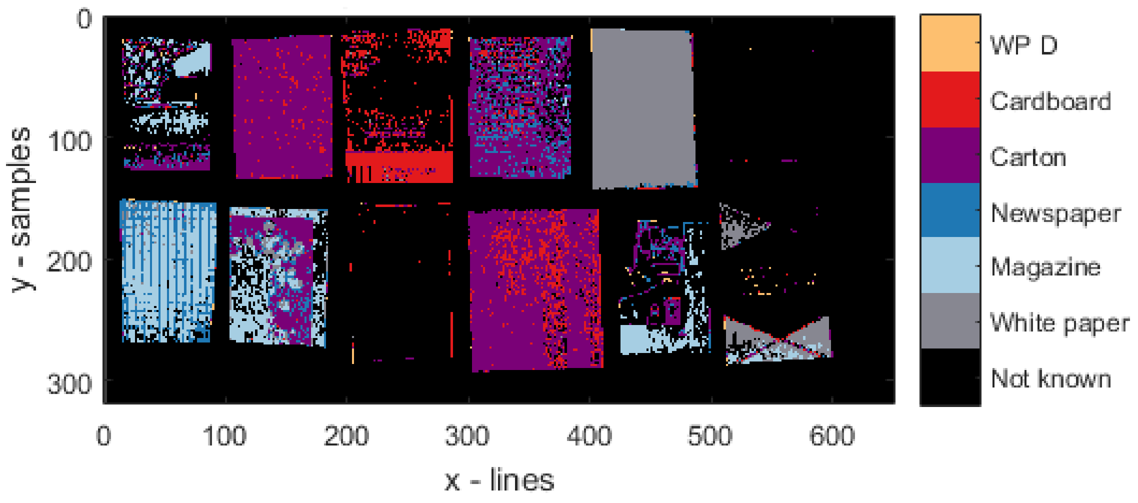


Figure 4.24 Predicted classes by the hierarchical model of hypercube *Validation 1*, strict classification

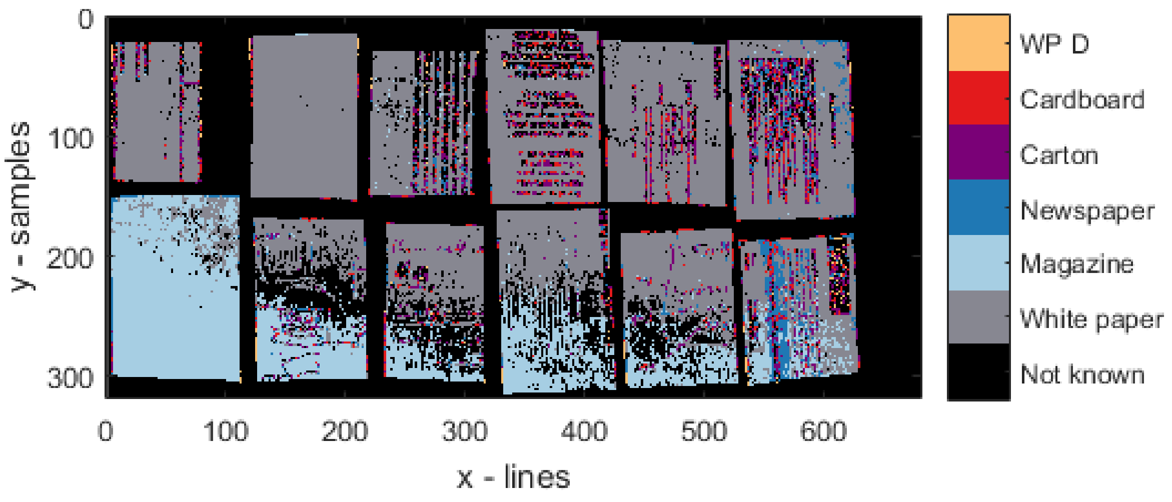


Figure 4.25 Example of how illumination affected classification (hypercube *WP P 13-24*)

To improve the quality of the classification and to reject useless information, a simple generalization model was developed on top of the pixel-by-pixel classification. For the generalization both the pixel-by-pixel classification image and the raw hypercube were used. For the generalization a few assumptions were made:

- that each sample always belongs to only one class
- that the samples are separated from each other spatially (no touching edges)
- that the samples were rectangles (or at least polygonal)

All of these assumptions can be made for paper waste as well, with some exceptions but the exception cases were not taken in to account in this model anyway. Figure 4.26 shows how the generalization process worked.

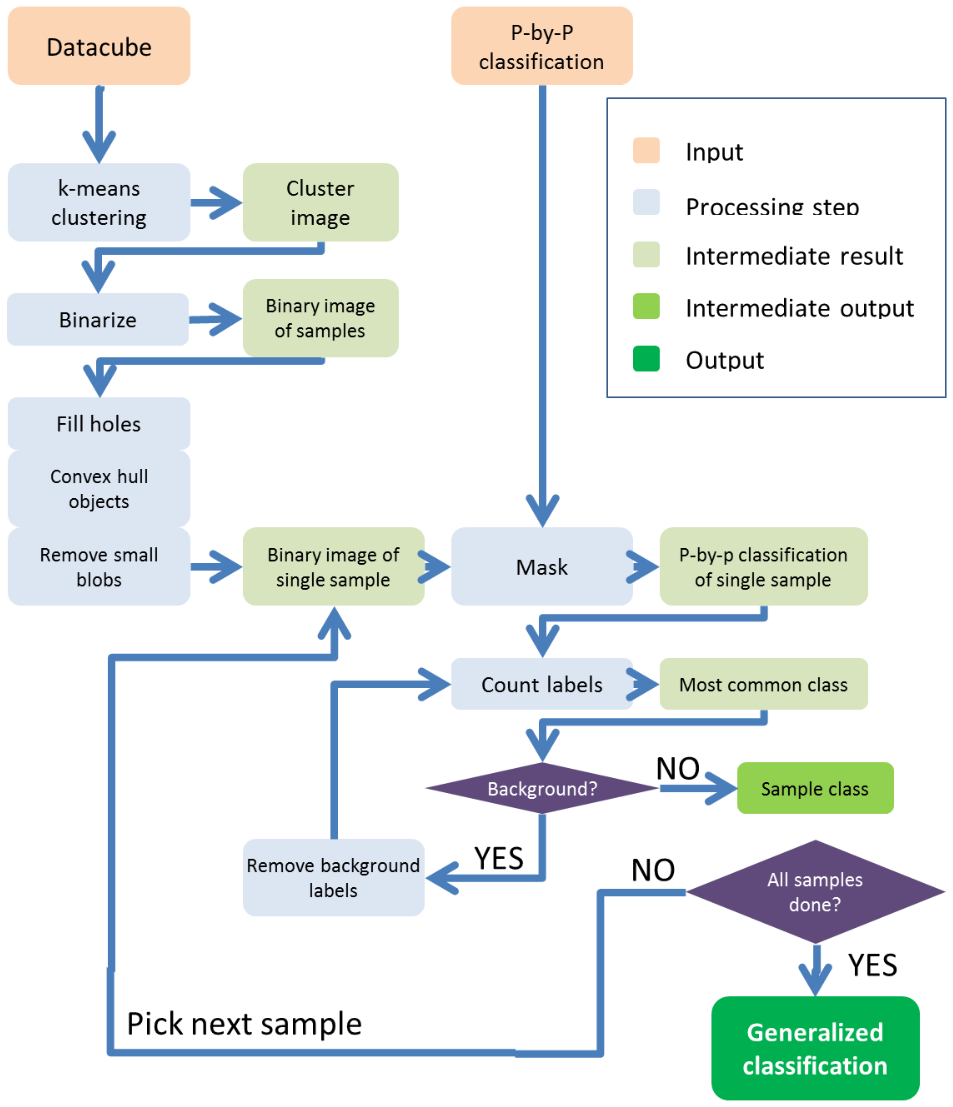


Figure 4.26 Generalization of the pixel-by-pixel classification

The hypercube was processed during reading from file with a small median filter that was applied on the cube to get rid of sensor artifacts (discussed in Chapter 4). Then, the hypercube was run through the k-means clustering (using three clusters, as this was the best solution for samples with print). The result was an image where the background was separated from the samples. This k-means result was binarized so that the result was a binary mask where zero signified background and ones signified sample pixels. The binary image was processed with simple MatLab machine vision algorithms:

- fill the holes left by the dark print and that were classified to the background class in k-means clustering (4 on Figure 4.27);
- remove small objects (skipped on Figure 4.27, there were no small blobs to remove);
- using convex hull on the sample to unite the samples that were divided by the dark print (5 on Figure 4.27);

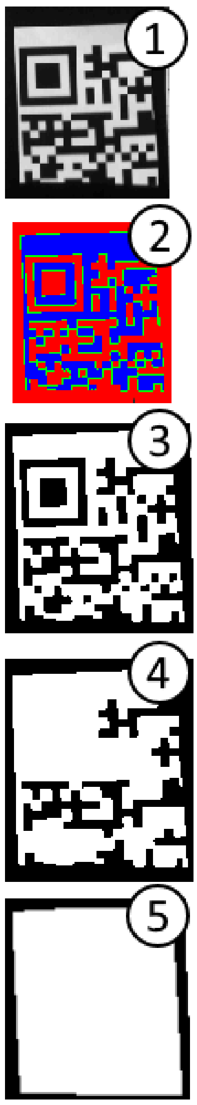


Figure 4.27 Example of processing. 1. Raw cropped image of hypercube at wavelength 1320 nm. 2. K-means clustering with 3 clusters. Backgroung is always assigned to 0. 3. Binarized k-means result. 4. Fill holes. 5. Convex hull of the object.

Figure 4.27 shows an example of an extreme case where the print caused most data points of the sample to be classified as background or N/A. It can be seen how the processing steps improve the area of the sample. Processing the hypercube this way allows the separation of samples to be much more accurate. An accurate positioning of samples is essential in physically sorting out the papers. As a result of these processing steps, a generalized classification is calculated where each sample is classified into one and only one class. An example of the generalized classification output can be seen on Figure 4.28, where the *Validation 1* hypercube was classified. The hypercube is the same as on Figure 4.24. When comparing the pixel-by-pixel classification and the generalized classification, it can be seen that each sample is classified into only one class. When there are not enough classified pixels within the sample (10% of all pixels) then the classification fails, showing that the class is not known. The samples are labeled by class and their centroid is shown. The centroid could be used for giving spatial information to the physical sorting machine.

Figure 4.28 also shows a case where the k-means clustering failed to correctly separate a sample. The sample at the far right of the bottom row of Figure 4.28 is from an advertisement that has a glossy and thick material, much like a magazine cover. The advertisement has a very dark printed middle part (see Figure 4.24) that was not detected by the k-means clustering. Therefore the lighter parts are considered as separate samples. To fix this issue, a localized threshold could be used alongside with the k-means clustering to adapt to the dark black samples and to separate them from the background. However, improving the sample detection method was outside the scope of this work and should be done in the future.

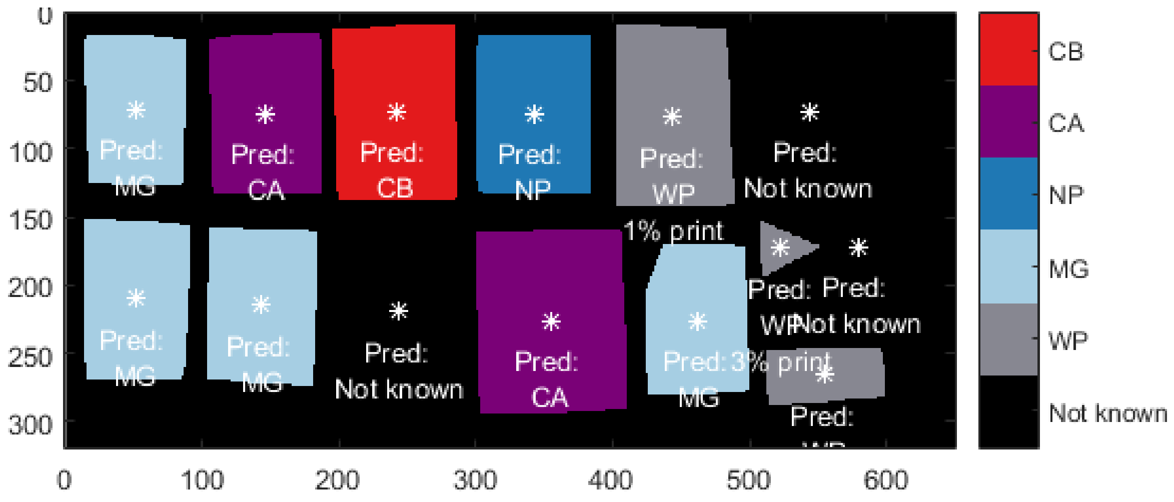


Figure 4.28 Generalized classification of hypercube *Validation 1*

Figure 4.29 shows that if a sample is classified as white paper, the percentage of coverage is calculated and displayed as well. This helps to sort the white paper further as required in Table 2.6. The percentage calculation was done using a rigid thresholding of the sample area, strictly inside the sample area (meaning no background data points were involved). As white paper has fairly constant reflectance values (and recorded brightness values) then the threshold was chosen to be 60% of the maximum brightness of the image.

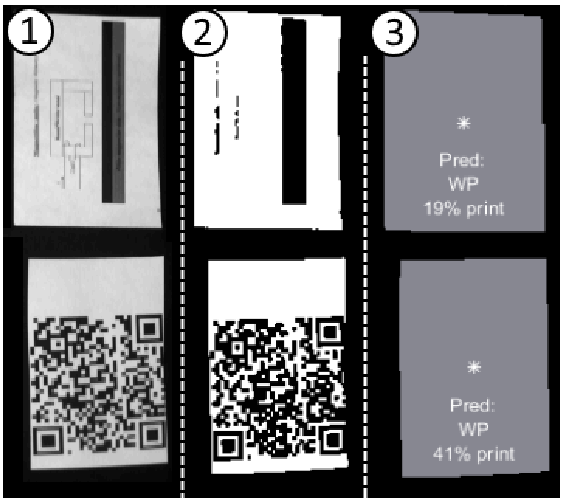


Figure 4.29 Computation of the percentage of areas covered in print.

Wavelength band 22 (1320 nm) was chosen for the thresholding image. The result of the thresholded image is a binary 2D mask containing the value „0“ for areas covered with print and „1“ for clean areas. To get the percentage, the dark pixels were counted and divided by the whole area of the sample. Figure 4.29 shows the steps of the percentage calculation. This percentage only includes dark print to the printed pixels. For example, if the page was covered in bright yellow print then the percentage would still very likely show around 0%. One possible solution to include light prints in the percentage calculation would be to incorporate a simple RGB-camera to the system that would detect colors (see more in Chapter 6).

All in all, the generalization model applied on top of the pixel-by-pixel classification improved the quality of the classification greatly (see more in Chapter 5) as well as provided the system with outputs that could be passed on to other parts of the system for further processing and physical sorting.

4.3.5. Validation method

The previous chapter described how the generalized classification method operates and how to provide an output from the classification system. Before the system can be applied on real-time data, however, it must be validated to show the quality and accuracy of the proposed classification method. For this, a few MatLab scripts were developed that provided both numerical and visual information on the quality of the model.

The samples on the hypercubes that were used for validation were manually classified using the same method as for the model calibration part. Then the hypercubes were classified using the methods described above. To get an estimation on how the method performed, the manually classified regions were compared with the generalized classification images (for a description of the validation script, see Appendix 7). For each hypercube, two error images were generated. The first was the errormap of the pixel-by-pixel classification, the second was the errormap of the generalized classification. The validation method compared all of the pixels inside the region of interest (ROI) that was manually selected. So if the classification method did not correctly detect the area of the sample then that also counted as an error alongside misclassification. For the pixel-by-pixel errormap, the accuracy percentage of each sample was given as an output (see file *Validation percentages.xml* for the single sample percentages and Chapter 5 for the average validation hypercube percentages). For the generalized errormap a binary array was provided that showed if a specific sample was correct or not. Examples of errormaps are shown on Figures 4.30 and 4.31, generated of the *Validation 1* hypercube. On the figures, the centroid of the sample can be seen as well as the real class provided by the manual classification. If the area is green then that means that it is incorrectly classified.

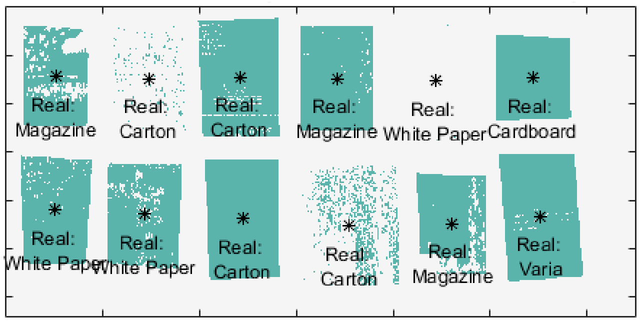


Figure 4.30 Pixel-by-pixel errormap of hypercube *Validation 1*. Green areas are misclassified

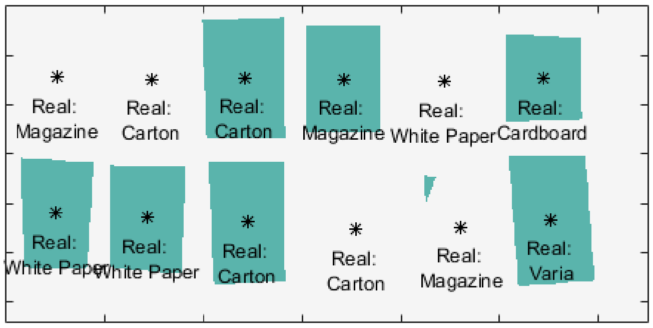


Figure 4.31 Generalized errormap of hypercube *Validation 1*. Green areas are misclassified

5. VALIDATION AND CLASSIFICATION RESULTS

Three PLS-DA classification models were combined into one hierarchical model. The result was a numeric output of each pixel's classification. Additionally to pixel-by-pixel classification (and based on pixel-by-pixel classification), a generalized classification model was developed that in the case of white paper also output the percentage of paper covered with print. The developed scripts used the models generated in the PLS_Toolbox to suit the hypercubes generated by Resonon cameras and to show the output as images. The more interesting aspects of the classification are discussed below, all of the validation images are in the folder *Matlab data/NIR/Validation Results* and the scripts can be run to classify arbitrary Resonon hypercubes (see Appendix 7 for overview of the scripts).

For validation of the classification model, ten hypercubes were used. For each hypercube classified ROIs were assigned. Three of the validation images contained validation samples that were not in any of the training datasets and the materials were randomly selected. Some of the validation samples were from classes that were not present at all in the training set. On the other hand, the remaining seven hypercubes were of the classes included in the training set and some of the samples in the hypercubes were used in the training set. Nevertheless, only single pixels from each sample were used for the training, and the hypercubes include hundreds of thousands of pixels so it was no problem that the validation set included some of the training set data. Table 5.1 shows the results of the validation.

Table 5.1 Accuracy of the classification model

	Pixel-by-pixel model accuracy, %			Generalized model, accuracy, %		
	Per hypercube	Average		Per hypercube	Average	
Validation 1	27,5			33,3		
Validation 2	41,4	27,6		50,0	32,9	
Validation 3	13,9			15,4		
WP 1-10	99,1			100,0		
WP G 1-12	99,1			100,0		
WP P 13-24	56,7		56,1	83,3		66,2
MGS 25-36	63,5	68,3		83,3	82,9	
CA 13-24	84,7			91,7		
CB 1-12	48,3			63,6		
NP 37-48	27,1			58,3		

As can be seen in Table 5.1, the classification accuracy for the three validation hypercubes was very low, only 27,6% on average (the per-sample validation results can be seen in *Validation percentages.xmlx*). As mentioned before, the validation sets contained many classes that were not present in the training data nor in the class labels' set. The model showed much higher accuracy when run on the distinct class materials. Table 5.1 also shows how much the generalized prediction improved the classification accuracy. For the validation sets the improvement was not very high but for the other tested hypercubes the improvement was significant, especially for the magazine and newspaper samples due to the fact that they were prone to misclassification into the carton class. Darker areas of materials very often were misclassified as carton, probably due to the smaller total intensity of the carton spectra. Figure 5.1 shows how newspaper samples are very likely to be classified as carton. For the generalized classification, as specialized part of the script checked if carton was the most numerous class within the sample and if so, whether there was a certain percentage of newspaper classification also present. In that case the newspaper class was preferred.

White paper classification was the focus point of this work due to the highest economic benefit (highest price on the recycling market of the waste paper materials). Therefore the print percentage is also included, to provide an even better classification of white paper. As seen in Table 5.1 and on Figure 5.2, the classification of white paper was very successful, reaching 100% for clean white paper samples, using the generalized classification method. However, as seen on the hypercubes *Validation 1-3* then the white paper is not always correctly classified. In the case of *Validation 3* in Figure 5.4 it can be seen that a sample of white paper is classified as cardboard. White paper samples being included in the cardboard stream is not a very big loss, but the opposite (and much more undesired) case can also occur. On Figure 5.4 (a.) it can be seen that a piece of advertisement leaflet was classified as white paper when it should have been classified as magazine or N/A. This misclassification was partially caused by the fact that the advertisement material was not in the training set but for future developments it might be beneficial to impose more strict classification rules on white paper. Another case where the classification was unsuccessful (and not only for white paper) was the case when the paper sheets were stacked on top of each other, as they naturally would be also in the waste paper stream. This was caused by the fact that much less light is reflected back from single sheets than from stacks of material. These cases should be investigated further.

Sorting out clean carton and cardboard could also be deemed successful, with respective accuracy percentages of 91,7% and 63,6% for the generalized model. The cardboard accuracy percentage is lower due to the fact that cardboard often got misclassified as carton. But because carton and cardboard were grouped together in Table 2.6 then mixing between these classes is not a very serious issue. However, the model definitely needs improvement in separating cardboard or carton that has paint on it from the waste paper stream. For example in hypercube *Validation 1* (see Figures 4.23, 4.24, 4.28, 4.30 and 4.31) in the top row, far right corner is a sample of cardboard with a top coating of black paint. This sample was picked up by the k-means clustering but was not classified by the model, probably due to lack of similar spectra in the training set.

On Figure 5.4 (b.) the carton sample had a thick black plastic triangle on top of the sample. The plastic part was not classified into any of the paper classes, as the pixel-by-pixel classification shows. Nevertheless, since the plastic was in the middle of the sample then the convex hull removed the cavity caused by the non-classified pixels and the fact that there was plastic on the sample did not carry on to the generalized classification. Figure 5.3 shows how non-paper materials were classified (the k-means image and pixel-by-pixel classification image are in the folder *Validation Results* in the accompanying files). In the figure, all samples are non-paper. Most of them were not identified as paper materials, with three exceptions, all of which were classified as carton. The materials that were classified as carton were rock, metal and plywood. In general, the model should be improved to detect more common non-paper materials as well in the waste paper stream to be able to sort them out.

Table 2.6 shows that the highest priority materials to be sorted out are white paper with varying degrees of print coverage, newspaper and carton/cardboard. All of these goals were reached with a high percentage of accuracy in separating all of these classes from the waste stream. Furthermore, a model to separate cardboard from carton was also included in the overall sorting model, however the success rate of this separation was not very high. There are numerous class separations that could be improved on, for example classifying magazine covers or stacks of materials reliably. Mentioned above were the most interesting results and conclusions that could be drawn from the validation process. For the full set of validation figures and information, refer to the folder *Validation Results* in the accompanying data.

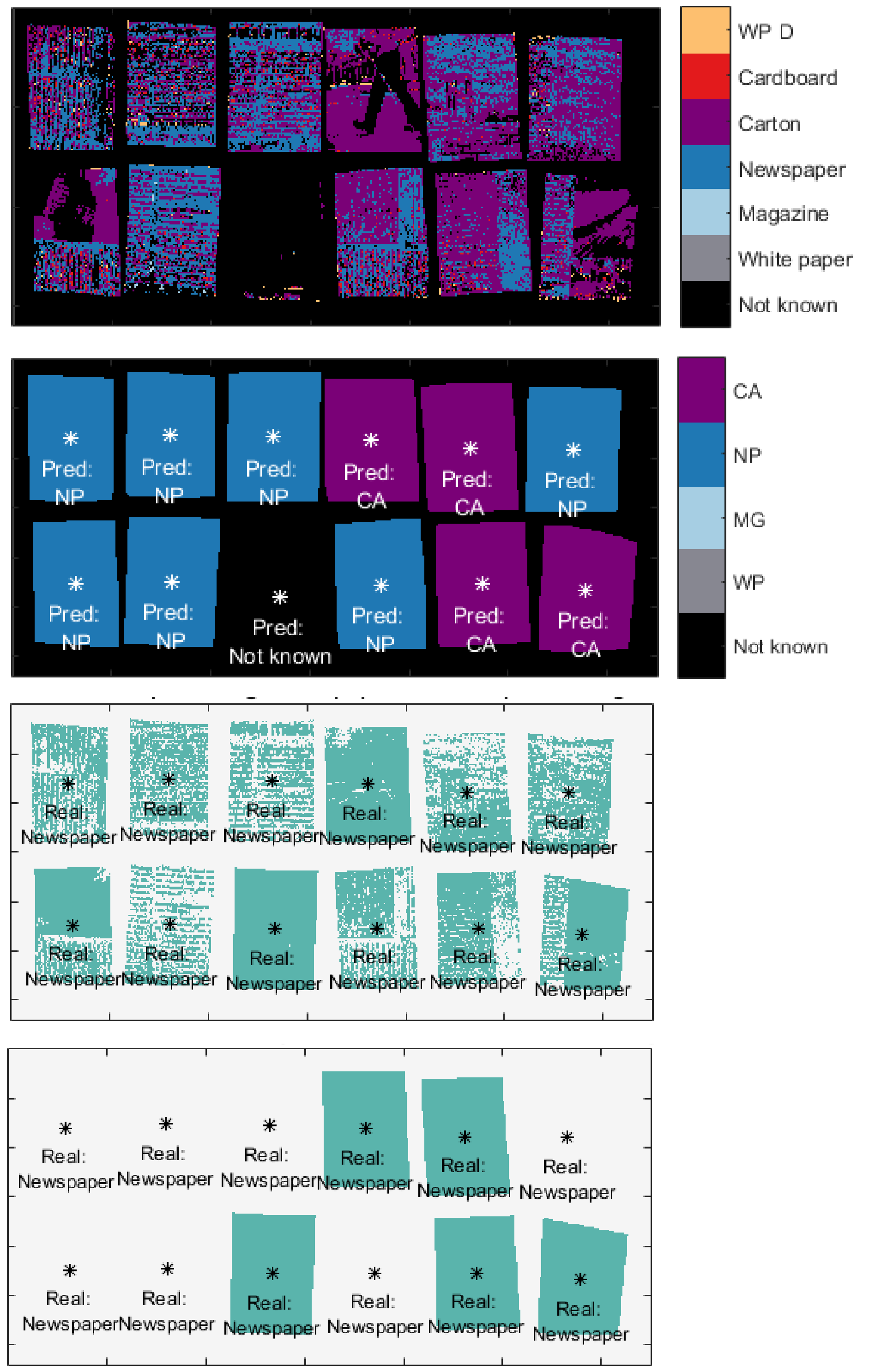


Figure 5.1 Validation of newspaper samples (hypercube *NPS 37-38*). From the top: the strict pixel-by-pixel classification; generalized classification; pixel-by-pixel errormap; generalized errormap.

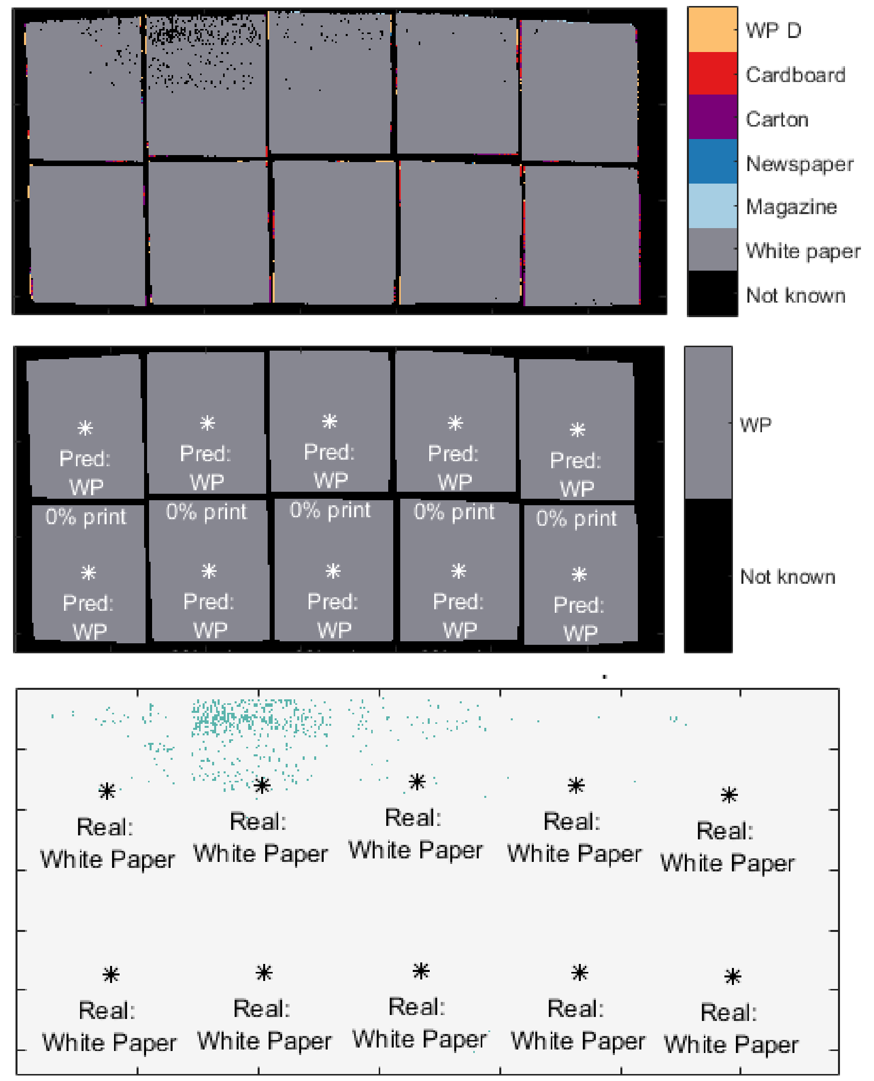


Figure 5.2 Validation results of white paper (hypercube *WP 1-10*). From the top: the strict pixel-by-pixel classification; generalized classification; pixel-by-pixel errormap. The generalized errormap is omitted because all samples were classified correctly. On the generalized classification figure the print coverage percentages were unfortunately cropped from the figure, but they were the same as for the top row: all zeros.

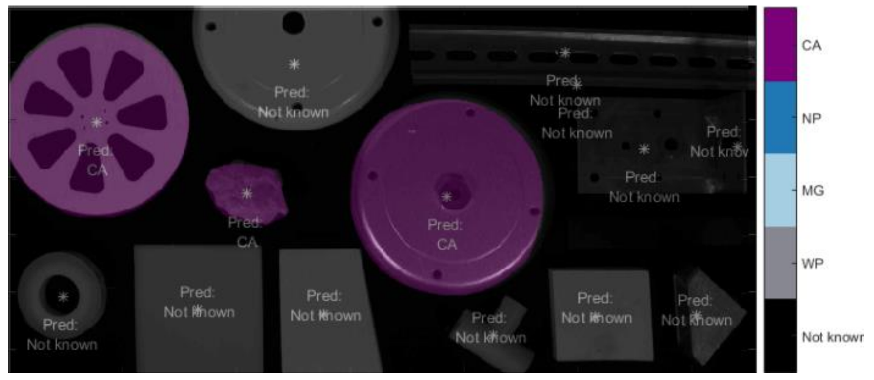


Figure 5.3 Non-paper materials. Image at wavelength 1320 nm overlaid with the generalized classification result.

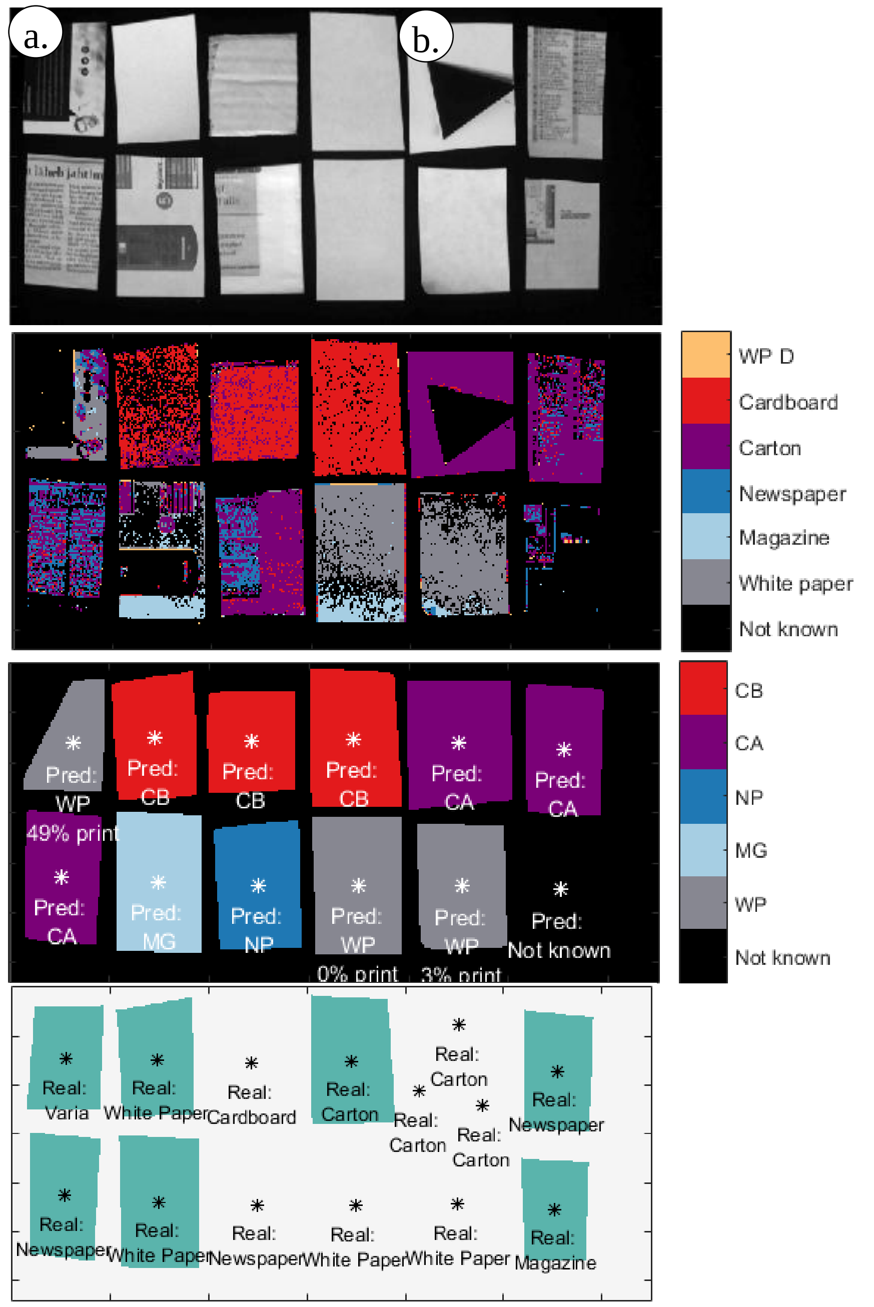


Figure 5.4 Results for *Validation 2*. From the top: cropped image of the hypercube at wavelength 1320 nm; the strict pixel-by-pixel classification; generalized classification; generalized errormap.

6. FUTURE GOALS

As seen above in the classification results chapter, the classification model is far from perfect. In this chapter an overview of the future possibilities is given, both on how to improve the model and how to make use of the information acquired in this work.

The first improvement could be applied to the data acquisition step. In this work, the camera was mounted on a moving head and the samples stayed stationary as the camera scanned them. This solution was suitable for the scope of the thesis but would never be used in an industrial setting so tests should be carried out with a stationary linescan camera and a linear conveyor belt moving the samples. Not only would this remove the spatial distortions associated with the changing of the working distance but it would also provide a much smaller field of view and therefore also a much more uniform lighting. It would also be possible then to improve the lighting intensity which would be especially beneficial for the Pika II data acquisition as it was much less light sensitive than the NIR camera. Furthermore, if it is possible to get less noisy results with the Pika II data than in this thesis, then combining the hypercubes and repeating the analysis would hopefully provide even better results.

Another possibility to improve the classification model would be the increase the number of classes included in the training set. For example, classes of highly glossy advertisement paper or painted cardboard could be added. Additionally, this work did not concentrate on the differences between single sheets of sample materials and stacks of samples. In practice, newspapers and magazines would rarely run through the waste paper stream as single sheets and therefore this situation should be thoroughly investigated. The possibility of foreign materials occurring in the waste paper stream should also be investigated further. The model proposed here did not classify many non-paper materials as paper, but there were some cases where it did. There was also a case (discussed in Chapter 5) where a plastic bit was not classified as paper by the pixel-by-pixel classification but the generalized model nevertheless classified it as paper. Therefore, improvements should be made to the generalized model, to detect foreign objects.

Another option to improve the model would be to omit the Pika II and use a simple machine vision CCD camera for color and print detection. The hyperspectral classification model was not capable of separating materials by color, for example gray and brown carton, or light print on white paper. The machine vision camera can be combined with the hyperspectral system to improve the quality of the model. The machine vision camera could also handle the print percentage calculation, reducing the workload on the hyperspectral data. This could improve speed.

As classification speed is definitely an important factor in sorting applications then multispectral cameras could be used for the classification, as they are faster due to the smaller number of wavelength bands. For example, one manufacturer of custom visible, NIR and SWIR multispectral cameras is Fluxdata. They provide multispectral cameras with three wavelength bands and promising a highly customizable range of wavelengths. However, a price quotation given by the sales representative of Fluxdata showed that the cameras are extremely expensive (the SWIR products ranging from 125000 USD to 145 000 USD each). Therefore a comparison of multispectral and hyperspectral cameras should be made, both in the applicability of multispectral cameras (whether the specific wavelengths can truly be used for classification) and the gain in speed/processing time. Furthermore, in this work, the analysis was done offline, meaning that the data was processed after it was acquired and no output was given for further processing. But in inline systems in the industry, the system should provide the output of class and spatial position online. Further work is required to assess the possible speeds at which this could be done in an industrial setting.

This work could be a good basis for further research on the topic. Many improvements can be made so that in the future could lead to a sorting system with numerous output streams of waste paper classes. When combined with a physical sorting system, a fully functional paper sorting machine could be developed which would be very attractive for many waste processing companies.

7. SUMMARY

Waste management is a very important part of modern society and in order to keep economies sustainable, recycling must be performed to reduce strain on the environment. This thesis concentrates on automated waste paper sorting. Currently waste paper is being sorted manually – if at all – and it is a very slow process with small throughput. Automated sorting of paper would increase the speed and reliability of the output streams and would motivate more companies to recycle due to the higher market value of the white long-fibered paper sorted out from the waste paper stream. This work was a continuation of the previous research in the Department of Mechatronics [14] where it was established that hyperspectral imaging can be used in automated waste paper sorting into numerous classes based on the samples' spectral response. The goals of this work were to collect a database of paper materials and their spectral responses as hypercubes; analyze the hypercubes to see what information they contained; and propose a classification model for waste paper classification.

Research in the field of material sorting using hyperspectral imaging (HSI) revealed that although it is used quite widely in agriculture, medicine and food production, HSI is not commonly used for waste sorting. If waste paper sorting is performed automatically, then usually RGB- or NIR wideband cameras are used. However, these solutions are limited to only a few classes (white paper vs. cardboard, for example). One research group that focused on waste paper sorting using HSI [12], found that it is very effective for this application. Based on the research, data analysis methods as well as classification methods were chosen for this work as well. The main goal in classification was to separate white paper (printer paper) from other papers, as it has the highest value on the recycling market. If the white paper was covered in print then the percentage of print coverage was to be given as output as well. Additionally, newspaper and cardboard were required to be separated from the waste paper stream.

The experiments were conducted on over 500 samples of paper materials and using two hyperspectral cameras provided by the Chair of Mechatronics Systems: Resonon Pika II (range 400 – 900 nm) and Resonon Pika NIR (range 900 – 1700 nm). The analysis of the acquired hypercubes showed that the 900-1700 nm range includes more relevant information for material classification than the 400-900 nm range. The latter range could successfully be used for material color detection and, if needed, provided information on which papers had glossy surfaces. However the Pika II data was not used for classification in the scope of this thesis, as the NIR hypercubes included even more information. The principal components analysis showed that for the NIR data, the most significant variances between classes occur between six wavelength bands. However, all bands were used in this work for classification purposes to guarantee maximal separation of classes.

For classification, three methods were tried: Support Vector Machines (SVM) discrimination, k-nearest neighbor (k-NN) classification and Partial Least-Squares Discriminant Analysis (PLS-DA). The classification models were built using Eigenvector's PLS_Toolbox [23]. PLS-DA proved to be faster and more informative than the other methods so this method was chosen for classification of paper waste. As it turned out, all the classes could not be separated with just one classification model. So a hierarchical classification model was built to first separate cardboard and carton from the other classes and then two more classification models that separated the other classes. In total, the proposed model could separate between five classes: white paper, magazine, newspaper, cardboard and carton. Two classification types were proposed: a pixel-by-pixel classification that classified each pixel of the image independently from its surroundings; and a generalized classification that assigned one class to the whole sample. The generalized model used k-means clustering as well as the pixel-by-pixel classification to assign classes. The generalized model was much more accurate in classification as was seen during the validation process of the classification methods. For validation, ten hypercubes were used, three of which contained paper materials that were not in the training set, and seven hypercubes that contained samples from the training set's classes. The highest accuracy was achieved while classifying white paper, the lowest for classification of random materials not in the training set.

In conclusion, the work done in this thesis can be deemed successful, as all goals were reached. Additionally, the work done for this thesis will stay in the Department of Mechatronics and further research can be based on it. Especially useful could be the database of papers and their respective hypercubes, as they could also be used as training material. The classification model could be improved by adding color information from the Pika II or a RGB-camera and more classes could be added. This work showed that class separation using HSI is possible and that HSI has a lot of information that can be put to use in waste paper treatment.

8. KOKKUVÕTE

Jäätmetöölus on tänapäeva ühiskonnas äärmiselt oluline ning selleks, et tagada jätkusuutlikku majanduslikku arengut, tuleb jäätmeid taaskasutada. Käesolev töö keskendub automaatsele jäätmepaberi sorteerimisele hüperspekraalkaamerat kasutades. Enamus tänapäeva jäätmetöötlusjaamu sorteerivad paberit käsitsi – kui üldse – ning see on väga aeganõudev protsess, mille kasutegur on väike. Automaatne papberi sorteerimine teeks kogu protsessi kiiremaks ja täpsemaks ning sorteeritud paberi (eriti valge paberi) kõrgem hind vanapaberi turul motiveeriks paljusid firmasid taaskäsitlusega tegelema. Käesolev töö jätkas Mehaatroonikainstituudi varasemat uurimistööd [14], milles instituudiväliste mõõtmistega tehti kindlaks, et paberi sorteerimine hüperspektraalkaameratega on võimalik. Antud töö eesmärgid olid: paber- ja pappmaterjalide andmebaasi kogumine ja nendest hüperspektraalsete andmete kogumine; hüperspektraalsete andmete analüüs, et välja selgitada, milliseid andmeid on neist võimalik välja lugeda; ning välja pakkuda klassifitseerimismudel paberi klasside eraldamiseks.

Varasemad tööd materjalide sorteerimisest kasutades hüperspekraalkaameraid näitasid, et kuigi hüperspektraalset infot kasutatakse küllalt laialdaselt põllumajanduses, meditsiinis ja toiduainetööstuses, siis jäätmekäitluses seda eriti ei kasutata. Kui paberjäätmeid automaatselt sorteeritakse, siis pigem RGB- ja/või NIR-lairibakaameratega, mis suudavad jäätmed sorteerida vaid kahte-kolme klassi. Üks uurimisgrupp, kes keskendusid paberjäätmete sorteerimisele hüperspektraalkaameraid kasutades [12], leidis, et antud lahendus on väga efektiivne. Uurimistöö põhjal valiti välja eelnevalt edukad meetodid hüperspektraalsete andmete töötlemiseks ja klassifitseerimiseks ning rakendati neid ka käesoleva töö käigus saadud andmetele. Antud töö põhiülesandeks sai valge paperi (printeripaberi) eraldamine muust materjalist, kuna sellel on vanapaberi turul kõrgeim väärtus. Kui valge paber oli kaetud trükiga, siis lisaks klassile andis väljund ka trükiga kaetuse protsendi. Lisaks valgele paberile keskendus käesolev meetod ka ajalehe- ja ajakirjapaberi ning papi ja kartongi eraldamisele.

Mõõtmised viidi läbi üle 500 paber- ja pappmatrjali näidisega ning kasutades kahte Mehaatroonikainstituudis asuvat hüperspektraalkaamerat: Resonon Pika II (lainepikkuste vahemikus 400 – 900 nm) ja Resonon Pika NIR (lainepikkuste vahemikus 900 – 1700 nm). Mõõtmiste käigus saadud andmete analüüs näitas, et lainepikkuste vahemik 900 – 1700 nm sisaldas rohkem klassifitseerimiseks olulist infot, kui lainepikkuste vahemik 400 – 900 nm. Viimane vahemik näitas materjalide värviinfot ja võimaldas tuvastada läikega pindu. Sellegipoolest ei kasutatud lainepikkuste vahemikku 400 – 900 nm käesoleva töö klassifitseerimismudelites, kuna vahemik 900 – 1700 nm sisaldas rohkem infot. Andmete PC analüüs (PCA) näitas, et kogu lainepikkuste vahemikust on võimalik eraldada kuus lainepikkust, mis sisaldavad enim erinevusi klasside vahel. Sellegipoolest kasutati käesoleva töö klassifitseerimisel kõiki lainepikkusi, et tagada maksimaalne klasside eristuvus ja näidata hüperspektraalse klassifitseerimise võimalusi.

Klassiftseerimiseks prooviti kolme meetodit: *Support Vector Machines*'i (SVM) ja *k-nearest neighbor*'i (k-NN) klassifitseerimist ning *Partial Least-Squares Discriminant Analysis*'i (PLS-DA). Klassifitseerimismudelid genereeriti Eigenvector'i PLS_Toolbox'iga [23]. Kõige edukamaks klassifitseerimise meetodiks osutus PLS-DA, mis oli kiirem ja palju informatiivsem kui teised meetodid. Selgus, et paberi klasse ei saa eraldada vaid ühe klassifitseerimise mudeliga. Seega koostati hierarhiline klassifitseerimise mudel, mis sisaldas kolme mudelit. Esmalt eraldati papp ja kartong muust pabermaterjalist. Seejärel eraldasid kaks mudelit materjali viieks klassiks: valge paber, ajakiri, ajaleht, papp ja kartong. Välja pakuti kaks klassifitseerimise tüüpi: pikslipõhine klassifitseerimine, mis klassifitseeris iga piksli sõltumatult ümbritsevatest pikslitest; ning üldistatud klassifitseerimismudel, mis kasutas nii pikslipõhist klassifitseerimist kui ka *k-means clustering*'i, et määrata igale näidisele üks ja ainult üks klass. Üldistatud mudel oli täpsem, nagu oli näha klassifitseerimismudeli valideerimistulemustest. Mudeli valideerimiseks kasutati kümmet hüperspektraalpilti, millest kolm sisaldasid pabermaterjale, mille klasse mudeli kalibreerimisel ei kasutatud. Ülejäänud seitse hüperspektraalpilti sisaldasid näidiseid klassidest, millega mudel oli kalibreeritud. Suurim täpsus saavutati valge paberi klassifitseerimisel, madalaim aga nende materjalide klassifitseerimisel, mida kalibreerimishulgas ei olnud.

Kokkuvõtteks võib käesoleva töö lugeda edukaks, kuna kõik püstitatud eesmärgid saavutati. Lisaks jäävad kõik mõõtmiste ja analüüsi käigus saadud andmed Mehhatroonikainstituuti ning nende andmetega saab uurimistööd jätkata. Eriti võib kasu olla pabermaterjalide andmebaasist, kuna neid võib kasutada ka õppematerjalina. Klassifitseerimismudelit saab edasi arendada ning loodetavasti tulevikus ka töösse rakendada. Käesolev töö näitas, et hüperspektraalsed andmed võimaldavad materjale jaotada paljudeks eri klassideks.

9. REFERENCES

[1] Chang, C.-I. Hyperspectral Data Exploitation: Theory and Applications, Chapter 2 Hyperspectral Imaging Systems. Published online 2007

[2] Microimages, Inc. Introduction to Hyperspectral Imaging [PDF] http://www.microimages.com/documentation/Tutorials/hyprspec.pdf (18.02.2016)

[3] Amigo, J. M., Babamoradi, H., Elcoroaristizabal, S. Hyperspectral image analysis. A tutorial (2015) – *Analytica Chimica Acta.* Volume 896, pages 34–51

[4] Lu, G., Fei, B. Medical hyperspectral imaging: a review (2014) – *Journal of Biomedical Optics*. Volume 19, Issue 1, pages 1-23

[5] What is Imaging Spectroscopy (Hyperspectral Imaging)? [WWW] http://www.markelowitz.com/Hyperspectral.html (18.02.2016)

[6] Sacré, P.-Y., De Bleye, C., Chavez, P.-F., Netchacovitch, L., Hubert, Ph., Ziemons, E. Data processing of vibrational chemical imaging for pharmaceutical applications (2014) – *Journal of Pharmaceutical and Biomedical Analysis.* Volume 101, pages 123–140

[7] Mahesh, S., Jayas, D.S., Paliwal, J., White, N.D.G. Hyperspectral imaging to classify and monitor quality of agricultural materials (2015) – *Journal of Stored Products Research.* Volume 61, pages 17–26

[8] Moscetti, R., Saeys, W., Keresztes, J.C., Goodarzi, M., Cecchini, M., Danilo, M., Massantini, R. Hazelnut Quality Sorting Using High Dynamic Range Short-Wave Infrared Hyperspectral Imaging (2015) – *Food Bioprocess Technol*. Volume 8, pages 1593–1604

[9] Barnabé, P., Dislaire, G., Leroy, S., Pirard, E. Design and calibration of a two-camera (visible to near-infrared and short-wave infrared) hyperspectral acquisition system for the characterization of metallic alloys from the recycling industry (2015) – *Journal of Electronic imaging.* Volume 24, issue 6

[10] Picón, A., Ghita, O., Whelan, P.F., Iriondo, P.M. Fuzzy Spectral and Spatial Feature Integration for Classification of Nonferrous Materials in Hyperspectral Data (2009) - *IEEE Transactions on Industrial Informatics*. Volume. 5, number. 4, pages 483-494

[11] Picon, A., Ghita, O., Iriondo, P., Bereciartua, A., Whelan, P.F. Automation of waste recycling using hyperspectral image analysis (2010) - *Emerging Technologies and Factory Automation (ETFA), 2010 IEEE Conference on.* Pages 1-4

[12] Tatzer, P., Wolf, M., Panner, T. Industrial application for inline material sorting using hyperspectral imaging in the NIR range (2005) - *Real-Time Imaging*. Volume 11, pages 99 – 107

[13] Rasmus NJ. The VTTVIS line imaging spectrometer—principles, error, sources, and calibration. Denmark: Pitney Bowes Management, 2002

[14] Põlder, A., Juurma, M., Tamre, M. Waste Paper Sorting Using Imaging Spectroscopy (2013) - *Publication of Doctoral School of Energy and Geotechnology*, pages 283-284

[15] Marshall, J.L, Williams, P., Rheault, J.-P., Prochaska, t., Allen, R.D., DePoy, D. L. Characterization of the Reflectivity of Various Black Materials (2014) - *Ground-based and Airborne Instrumentation for Astronomy V*

[16] MatLab Recursive directory listing toolbox [WWW]: http://www.mathworks.com/matlabcentral/fileexchange/19550-recursive-directory-listing (23.04.16)

[17] MatLab Field Mapping Toolbox [WWW]: http://www.mathworks.com/matlabcentral/fileexchange/30853-field-mapping-toolbox/content/medfilt3.m (26.04.16)

[18] Paper and board – European list of standard grades of recovered paper and board for recycling. (2014). EVS-EN 643:2014. Tallinn: Estonian Centre For Standardisation.

[19] The new EN 643. CEN Standard [PDF] http://www.cepi.org/system/files/public/epw-presentations/2013/Standard+EN+643+-+new.pdf (3.04.2016)

[20] Randolph, T.W. Scale-based normalization of spectral data (2005) [PDF]: http://research.fhcrc.org/content/dam/stripe/randolph/files/Publications/Normalization_DiseaseMarkers_Offprint.pdf (10.05.2016)

[21] Worley, B., Powers, R. Multivariate Analysis in Metabolomics (2015) [WWW]: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4465187/ (10.05.2016)

[22] MatLab Pattern Recognition and Machine Learning Toolbox [WWW]: http://www.mathworks.com/matlabcentral/fileexchange/55826-pattern-recognition-and-machine-learning-toolbox (19.04.2016)

[23] Eigenvector PLS_Toolbox for MatLab [WWW]: http://eigenvector.com/ (22.04.16) and http://wiki.eigenvector.com/ (15.04.16)

APPENDICES

Appendix 1. Resonon Pika II datasheet

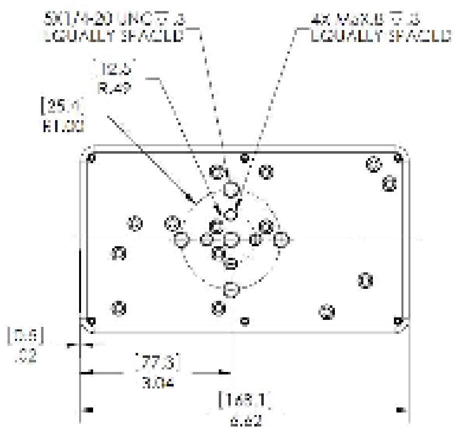
Pika II (400 – 900 nm)

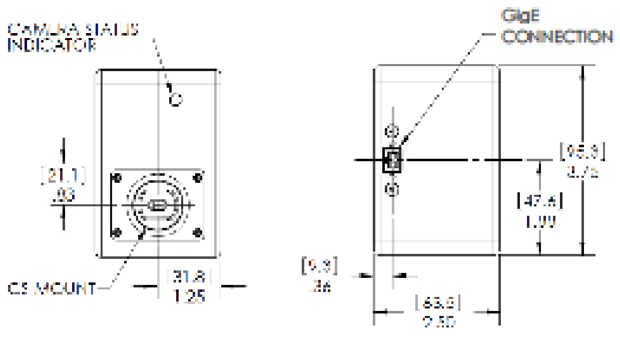
The **Pika II** is our most popular and most affordable hyperspectral imaging camera.

Pika II Specifications

Spectral Range	400 – 900 nm
Spectral Resolution *	2.1 nm
Spectral Channels	240
Spatial Channels	640
Max Frame Rate	145 fps
Bit Depth	12
Weight	2.8 lbs, 1.3 kg
Dimensions	9.7 x 16.8 x 6.4 cm
Connection Options	GigE
Temperature Range	46-90 F, 8-32 C
f/#	f / 3.0
Avg. RMS Spot Radius	7 µm
Smile (peak-to-peak)	5 µm
Keystone (peak-to-peak)	7 µm

* *The number of spectral channels equals the spectral range divided by the spectral resolution, and depends on the RMS spot size. The number of independent spectral channels is NOT the same as the number of sensor pixels in the spectral direction.*





www.resonon.com inquiry@resonon.com 1.406.586.3356

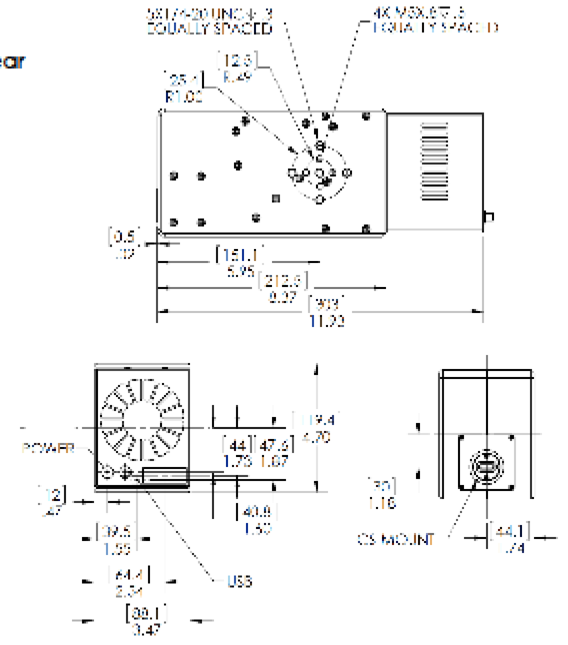
Appendix 2. Resonon Pika NIR datasheet

Pika NIR (900 – 1700 nm)

Our hyperspectral imaging camera covering the near infrared wavelengths.

Pika NIR Specifications

Spectral Range	900 – 1700 nm
Spectral Resolution *	5.5 nm
Spectral Channels	145
Spatial Channels	320
Max Frame Rate	100 fps
Bit Depth	14
Weight	10.4 lbs, 4.7 kg
Dimensions	11.9 x 30.5 x 8.9 cm
Connection Options	USB
Temperature Range	32-122 F, 0-50 C
f/#	f / 1.8
Avg. RMS Spot Radius	10 µm
Smile (peak-to-peak)	3 µm
Keystone (peak-to-peak)	5 µm



www.resonon.com inquiry@resonon.com 1.406.586.3356

Appendix 3 – Overview of the database of samples

The folders including the samples have files called Scan Info.txt, in which the order of all the samples on all the hypercubes is explained.

Table A.0.1 Database of samples

Class	**Subclass**	**Code**	**Count**	**Both sides?**
White paper WP	White printer paper	1-50	50	
	White thick paper	WP_40_THICK_oneSheet	1	
Yellow Paper Y	Watercolor paper	Y WC	9	
	Notepad	Y NP	8	
	Notepad perforated edges	Y NP Perf	6	
	Notepad recycled	Y NP R	4	
	Diploma	Y D	9	
	Sketch recycled	Y S R	10	
	Thicker print paper	Y PR	2	
White paper WP with print	Text, markings on one side	WP 1S	36	
	Text, markings on both sides	WP 2S	12	
	Graphic print, either 1S or 2S	WP G	12	
	Mostly covered in print, strong print	WP COV	36	
	Drafting paper (squares), empty	WP DR E	12	
	Drafting paper (squares), pen	WP DR P	12	
Magazine	Cover	MGC	12	Yes
	Sheets	MGS	36	
Newspaper	Matte pages	NPM	48	
	Shiny pages (Arguably same as MGS)	NPS	24	
Carton	Clean: Without any covering/paint	CAC	24	
	One side painted	CA1S	24	Yes
	Strong paper, one side painted	CASP	24	Yes
	Colored strong paper	CACO	8	
	Colored strong paper, both sides	CACO2	10	

Cardboard	One side covered in paint	CB1S	10	Yes
	Two sides covered in paint	CB2S	12	Yes
	Clean cardboard	CBC	24	
Advertisement	Shiny printed paper	AD	36	
Class	**Subclass**	**Marking**	**Count**	**Both sides?**
Checks	Checks and Receipts	CH	24	
Colored paper	Colored paper	COP	12	
	Post-its	PO	20	
WP with cover and fasteners	Plastic binder	Cover 1	1	
	Plastic cover	Cover 2	3	
	Plastic cover thick	Cover 3	1	
WP with tapes etc.	Tape	Tape	3	
	Rubber band	RBAND	1	
	String	String	3	
	Paperclip		1	
		Total	580	

Appendix 4 – Table of loadings for PCA model, NIR

Table A.2 Table of loadings, PCA, NIR

Wavelength, nm	PC1	Abs	% of PC1	PC2	Abs	% of PC2	PC3	Abs	% of PC3	PC4	Abs	% of PC4
961	-0,069	0,069	0,005	0,045	0,045	0,002	0,613	0,613	0,376	0,086	0,086	0,007
978	-0,050	0,050	0,002	0,033	0,033	0,001	0,460	0,460	0,212	0,061	0,061	0,004
994	-0,037	0,037	0,001	0,020	0,020	0,000	0,364	0,364	0,132	0,040	0,040	0,002
1010	-0,026	0,026	0,001	0,007	0,007	0,000	0,296	0,296	0,088	0,016	0,016	0,000
1027	-0,020	0,020	0,000	-0,007	0,007	0,000	0,243	0,243	0,059	-0,005	0,005	0,000
1043	-0,015	0,015	0,000	-0,021	0,021	0,000	0,202	0,202	0,041	-0,018	0,018	0,000
1059	-0,011	0,011	0,000	-0,031	0,031	0,001	0,176	0,176	0,031	-0,025	0,025	0,001
1076	-0,008	0,008	0,000	-0,040	0,040	0,002	0,155	0,155	0,024	-0,026	0,026	0,001
1092	0,005	0,005	0,000	-0,056	0,056	0,003	0,065	0,065	0,004	-0,033	0,033	0,001
1108	0,005	0,005	0,000	-0,054	0,054	0,003	0,049	0,049	0,002	-0,023	0,023	0,001
1125	0,005	0,005	0,000	-0,040	0,040	0,002	0,046	0,046	0,002	-0,003	0,003	0,000
1141	0,005	0,005	0,000	-0,020	0,020	0,000	0,042	0,042	0,002	0,021	0,021	0,000
1157	0,004	0,004	0,000	0,002	0,002	0,000	0,035	0,035	0,001	0,041	0,041	0,002
1174	0,004	0,004	0,000	0,026	0,026	0,001	0,027	0,027	0,001	0,055	0,055	0,003
1190	0,004	0,004	0,000	0,044	0,044	0,002	0,023	0,023	0,001	0,061	0,061	0,004
1206	0,004	0,004	0,000	0,051	0,051	0,003	0,015	0,015	0,000	0,051	0,051	0,003
1223	0,004	0,004	0,000	0,040	0,040	0,002	0,011	0,011	0,000	0,022	0,022	0,000
1239	0,005	0,005	0,000	0,007	0,007	0,000	0,013	0,013	0,000	-0,021	0,021	0,000
1255	0,005	0,005	0,000	-0,034	0,034	0,001	0,015	0,015	0,000	-0,068	0,068	0,005
1272	0,007	0,007	0,000	-0,078	0,078	0,006	0,020	0,020	0,000	-0,157	0,157	0,025
1288	0,008	0,008	0,000	-0,143	0,143	0,020	0,034	0,034	0,001	-0,245	0,245	0,060
1304	0,002	0,002	0,000	-0,227	0,227	0,052	0,037	0,037	0,001	-0,232	0,232	0,054

1321	-0,006	0,006	0,000	-0,282	0,282	0,079	0,035	0,035	0,001	-0,162	0,162	0,026
1337	-0,016	0,016	0,000	-0,290	0,290	0,084	0,026	0,026	0,001	-0,046	0,046	0,002
1353	-0,025	0,025	0,001	-0,259	0,259	0,067	0,017	0,017	0,000	0,094	0,094	0,009
1370	-0,030	0,030	0,001	-0,191	0,191	0,037	0,005	0,005	0,000	0,226	0,226	0,051
1386	-0,029	0,029	0,001	-0,089	0,089	0,008	-0,006	0,006	0,000	0,314	0,314	0,099
1402	-0,022	0,022	0,000	0,030	0,030	0,001	-0,019	0,019	0,000	0,348	0,348	0,121
1419	-0,012	0,012	0,000	0,139	0,139	0,019	-0,027	0,027	0,001	0,335	0,335	0,112
1435	0,000	0,000	0,000	0,228	0,228	0,052	-0,030	0,030	0,001	0,278	0,278	0,077
1451	0,012	0,012	0,000	0,287	0,287	0,082	-0,023	0,023	0,001	0,187	0,187	0,035
1468	0,024	0,024	0,001	0,318	0,318	0,101	-0,009	0,009	0,000	0,069	0,069	0,005
1484	0,037	0,037	0,001	0,335	0,335	0,112	0,004	0,004	0,000	-0,070	0,070	0,005
1500	0,048	0,048	0,002	0,320	0,320	0,103	0,018	0,018	0,000	-0,214	0,214	0,046
1517	0,052	0,052	0,003	0,261	0,261	0,068	0,037	0,037	0,001	-0,282	0,282	0,080
1533	0,067	0,067	0,004	0,183	0,183	0,033	0,034	0,034	0,001	-0,268	0,268	0,072
1549	0,154	0,154	0,024	0,103	0,103	0,011	0,041	0,041	0,002	-0,171	0,171	0,029
1566	0,173	0,173	0,030	0,075	0,075	0,006	0,042	0,042	0,002	-0,127	0,127	0,016
1582	0,199	0,199	0,040	0,048	0,048	0,002	0,048	0,048	0,002	-0,071	0,071	0,005
1598	0,238	0,238	0,057	0,034	0,034	0,001	0,048	0,048	0,002	-0,024	0,024	0,001
1615	0,293	0,293	0,086	0,028	0,028	0,001	0,045	0,045	0,002	-0,002	0,002	0,000
1631	0,369	0,369	0,136	0,004	0,004	0,000	0,033	0,033	0,001	0,020	0,020	0,000
1647	0,472	0,472	0,223	-0,060	0,060	0,004	0,029	0,029	0,001	0,070	0,070	0,005
1664	0,616	0,616	0,380	-0,169	0,169	0,029	0,028	0,028	0,001	0,181	0,181	0,033
	Sum		**1,000**	**Sum**		**1,000**	**Sum**		**1,000**	**Sum**		**1,000**

The table shows the loadings of variables (wavelengths) on four first PCs on the PCA model using only light parts of the samples. The columns % of PC show how many percent the PC is contributed by the variable. The variables with highest percentages are the most relevant in this PC and are marked with an orange background.

Appendix 5 – PCA of Pika II data

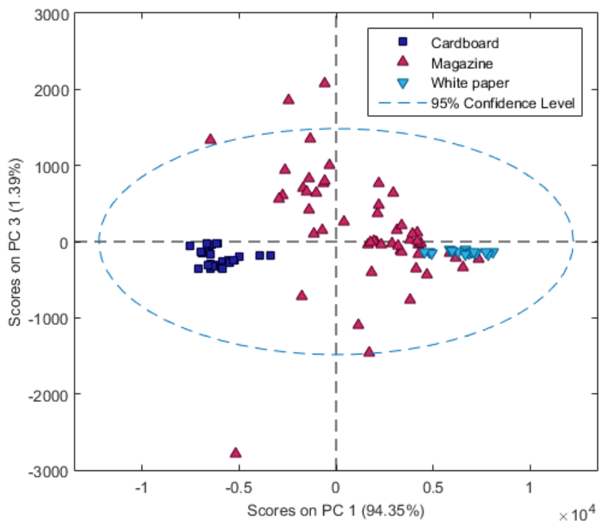


Figure A.0.1 Scores plot of PCA for Pika II data. PC1 vs PC3. Plot generated with PLS_Toolbox

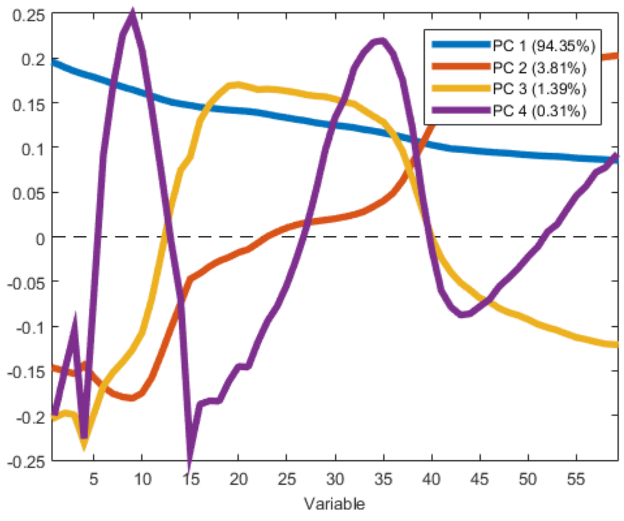


Figure A.0.2 Loadings plot of PCA for Pika II data. Generted with PLS_Toolbox

Appendix 6 – Overview of included data

Accompanying this work is a folder with the same name as the thesis. In the folder there is the following data:

1. The hypercubes acquired with both cameras in the folder *Hypercubes*. The hypercubes are divided into groups by class. For each class, recalibration was performed. In each folder for each group, there is a file called *Scan Info.txt* in which all of the samples are listed by position in the images.
2. The MatLab-generated files are in the folder *MatLab data*. This folder includes the saved manually classified specifications of ROIs (for both classification as well as validation) as well as the classification models used in this work. This folder also includes the validation results for the NIR camera.
3. The *MatLab scripts* folder includes all the scripts developed for the completion of this thesis. It also includes three toolboxes that were used to complete the tasks. This folder does not include the PLS_Toolbox files, however. So in order to run the classification models, the toolbox must be downloaded from [23].
4. The folder Research includes the scientific papers referenced in this work.
5. The folder Resonon includes the Resonon cameras' datasheets.
6. The root folder also includes three Excel files:
    a. The *Generalized model.xlsx* – the flowchart of the generalized model processing;
    b. The *NIR loadings.xlsx* – includes all the loadings data for the NIR PCA analysis;
    c. The *Paper database (23.04.16).xlsx* – includes the comprehensive list of all the paper samples.

Appendix 7 – MatLab scripts

Here described are the MatLab functions that can be used to work with the hypercubes as was done in the scope of this thesis. The most essential scripts' code is also presented, but the smaller non-important scripts are not given here as code (these can be seen in the folder *MatLab scripts*). Note that the Downloaded subfolder in the MatLab scripts folder must also be added to the MatLab path.

In the following table are descriptions of the scripts that could be written into the MatLab command line to execute the tasks described in the body text of this work.

Table A.3 MatLab command line functions. The scripts are presented in the order that they would naturally be used in.

1.	Syntax	Description
2.	`path = getPathAndAddSubfolders();`	Get the path of the MatLab scripts folder. It is needed for saving and opening ROIs
3.	`classEnumeration = addClassEnumeration(path, label, className)`	Add a class to the class enumeration file. If the file does not exist, it is created (see Appendix 7.1)
4.	`[hypercube, wavelengths, fileName] = readHypercubeFromFile (shouldDisplayImage, band)`	Read a hypercube from file into the MatLab workspace (is not required for the classification steps). See Appendix 7.2.
5.	`saveROIsOfCubes(path)`	Opens the graphical user interface to select hypercubes, assign ROIs on them and assign classes to these ROIs (see Appendix 7.3 and 7.4)
6.	`spectra = loadTrainingSpectra(path)`	Saves the spectra from the ROIs to file. This script can be modified to change the sample selection method. See Appendix 7.5
6.1	`[spArray, labelStrings, labelNumbers] = spectra2array(spectra)`	Converts the spectra format to three separate arrays. The spArray can be fed to the PLS_Toolbox as training data.
7.	`pls`	Opens the PLS_Toolbox. Needs the toolbox installed on PC
8.	`[predictionData, hypercube, fileName] = PLSDApredictHypercubeFromHierarchicalModel(modelIn)`	Opens the hypercube selector, and applies the hierarchical model on it. The model can be loaded into the workspace *\MatLab data\NIR\Calibrated Models\a_Hierarchical_Model.mat* (Appendix 7.6)
9.	`[generalizedPrediction, printPercentage] = getGeneralizedPrediction(predictionData, hypercube)`	Returns the generalized prediction from the pixel-by-pixel prediction. See Appendix 7.7

10.	`[predictionData, printPercentage, pixelByPixelValidation, generalizedValidation] = validateModel(model)`	Returns the same as 8. and 9., but also validates the hypercubes (if there are validation ROI files). Also returns the pixel-by-pixel and generalized model accuracy percentages. See Appendix 7.8

The table above and the following appendices do not include scripts called by the above scripts, for simplicity's sake. For the full list of the scripts, see the folder MatLab scripts, where all the scripts have been commented (about the inputs/outputs) for easy usage/modification.

NB! To use the classification models on the data, the PLS_Toolbox must be downloaded from [23]. The hypercubes can be opened, manually classified and the training data plotted without the PLS_Toolbox.

Appendix 7.1 `addClassEnumeration()`

```
% If a class enumeration file exists, the input class is added. If the file
% does not exist, it is created and then the class added.
% Input:
% path - path of the scripts folder
% label - integer, numerical label for the class
% className - string, label for the class
% Output:
% classEnumeration - a n x 2 cell array with the saved class labels

function classEnumeration = addClassEnumeration(path, label, className)
    fileName = 'classEnumeration';
    classEnumeration = openClassEnumerationFile(path);
    if ~isempty(classEnumeration{1,1})
        classEnumeration{size(classEnumeration, 1) + 1, 1} = label;
        classEnumeration{size(classEnumeration, 1), 2} = className;
    else
        classEnumeration{1, 1} = label;
        classEnumeration{1, 2} = className;
    end
    save([path, fileName], fileName, '-append');
end
```

Appendix 7.2 `readHypercubeFromFile()`

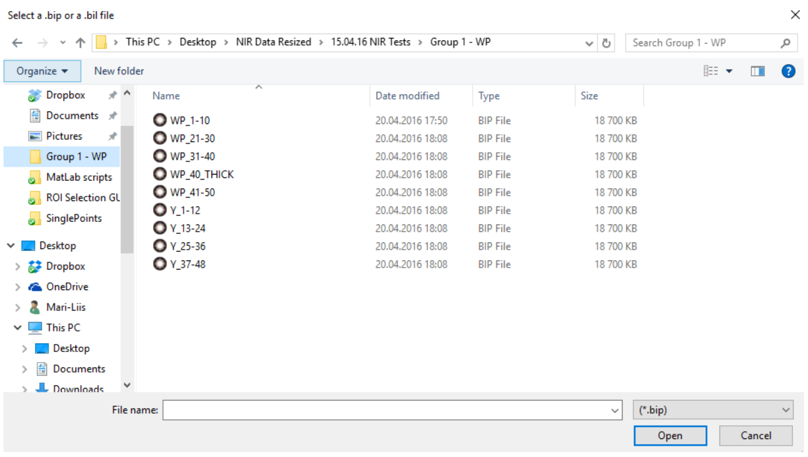
```
% Opens a file selector and reads selected hypercube into workspace
%
% Inputs:
%   shouldDisplayImage - boolean to indicate if user wants to see image
%   band - if shouldDisplayImage == true then this is the selected band
%
% Outputs:
%   hypercube - opened hypercube
%   wavelengths - an array of the central wavelengths of the bands
%   fileName - name of the hypercube file

function [hypercube, wavelengths, fileName] = readHypercubeFromFile(shouldDisplayImage, band)
    % Choose .bip or .bil file from computer
    [fileName, path] = uigetfile({'*.bip';'*.bil'}, 'Select a .bip or a .bil file');
    if fileName == 0
        hypercube = [];
        wavelengths = [];
        return
    end
    fullPath = fullfile(path, fileName);
    headerName = [fullPath, '.hdr'];
    fileName = strtok(fileName, '.');

    % Get file data from .hdr file
     % (see http://www.ehu.eus/ccwintco/uploads/d/dc/LoadHypercubesMatlab.pdf)
    disp(['Parsing header file ', headerName]);
    fileID = fopen(headerName,'r');
    C = textscan(fileID,'%s', 'delimiter','\n');
    strings = C{1,1};
    fclose(fileID);
     hypercubeData = getHeaderFileField(strings); % Parse header file data into struct
    wavelengths = hypercubeData.wavelengths;

    % Read hypercube
    disp('Reading hypercube...');
    inputHypercube = multibandread(fullPath, ...
      [hypercubeData.lines, hypercubeData.samples, hypercubeData.bands], ...
                                   hypercubeData.dataType, ...
                                   hypercubeData.offset, ...
                                   hypercubeData.interleave, ...
                                   hypercubeData.byteOrder);

    % Rotate hypercube (original orientation is wrong)
    hypercube = rot90(inputHypercube);

    if ~isempty(inputHypercube)
        disp('Finished reading hypercube...');
    end

    %Display image if requested of selected wavelength
    if shouldDisplayImage == true
        if band >= 0 && band <= hypercubeData.bands
            Y = squeeze(hypercube(:,:,band));
            figure('Name', (['Image at band ', num2str(band)]));
            imagesc(Y)
            colormap(gray)
            title(['Opened hypercube, image at band ', num2str(band)]);
            axis equal
            axis([0,hypercubeData.lines,0,hypercubeData.samples])
            xlabel('x - lines');
            ylabel('y - samples');
        else
            warndlg('Wavelength band out of range.', 'Could not create figure');
        end
    end
end
```

Appendix 7.3 `saveROIsOfCubes()` and `selectROIs()`

```
% Initializes the graphical user interface to select hypercubes,
% assign ROIs on them and assign classes to these ROIs
% Method of selection can be selected in Section Selection
%
% Input:
%   path - the path of the scripts folder
%
 function saveROIsOfCubes(path)
    roiDirectory = uigetdir(path, 'Select directory where to save the ROI files');
    roiDirectory = [roiDirectory, '\'];

    shouldStopFlag = false;

    while ~shouldStopFlag
        [hypercube, wavelengths, fileName] = readHypercubeFromFile(false, 40);

        selectROIs(path, roiDirectory, fileName, hypercube);

        questionDialog = questdlg('How would you like to continue?', ...
                    'Select action',...
                    'Select another cube', ...
                    'Finish', 'Finish');

        switch questionDialog
            case 'Select another cube'
                %continue on to selection
            case 'Finish'
                disp('Finished saving ROIs');
                shouldStopFlag = true;
                return
            otherwise
                disp('Finished saving ROIs');
                shouldStopFlag = true;
                return
        end
    end
end


% Opens the graphical user interface to
% assign ROIs on input hypercube and assign classes to these ROIs
%
% Input:
%   path - the path of the scripts folder
%   roiDirectory - directory of the ROI files
%   fileName - name of the hypercube
%   hypercube - the hypercube

function selectROIs(path, roiDirectory, fileName, hypercube)
     % Get image to display during ROI selection
     image = squeeze(hypercube(:,:,ceil(size(hypercube, 3) / 2)));

     shouldStopAddingFlag = false;

     if exist([roiDirectory, fileName, '.mat'], 'file')
        questionDialog = questdlg('The ROI file already exists for this cube
in this folder. Would you like to overwirte?', ...
                    'ROI file for cube already exists!',...
                    'Yes', ...
                    'No', 'No');
         switch questionDialog
             case 'Yes'
                 %continue on to selection
             case 'No'
                 disp(['Skipped selecting ROI for ', fileName]);
                 return
             otherwise
                 disp(['Skipped selecting ROI for ', fileName]);
                 return
         end
     end

     labelStrings = openClassEnumerationFile(path);
     labelNumbers = cell2mat(labelStrings(:,1));
     labelStrings = labelStrings(:,2);
     roiInfo = cell(1,1);
     visualization = zeros(size(hypercube,1), size(hypercube,2));
     count = 1;

     figure('Name', 'Selected areas')
     while ~shouldStopAddingFlag
        [selectedMask, xi, yi] = roipoly(mat2gray(image));

        [label,ok] = listdlg('PromptString','Select a label:',...
             'SelectionMode','single',...
             'ListString',labelStrings);

        if ok
            roiInfo{count, 1} = xi;
            roiInfo{count, 2} = yi;
            roiInfo{count, 3} = labelStrings(label);
            roiInfo{count, 4} = labelNumbers(label);

            visualization = visualization + selectedMask;
            imagesc(visualization);

            questionDialog = questdlg('Would you like to add ROIs or
save?',...
                    'How would you like to proceed?', ...
                    'Save', ...
                    'Select more', ...
                    'Cancel', 'Cancel');

            switch questionDialog
                    case 'Save'
                        save([roiDirectory, fileName, '.mat'], 'roiInfo');
                        shouldStopAddingFlag = true;
                        disp(['Saved to database file ', roiDirectory,
fileName, '.mat']);
                    case 'Select more'
                        % continue
                    case 'Cancel'
                        shouldStopAddingFlag = true;
                        return
                   otherwise
                        shouldStopAddingFlag = true;
                        return
            end
            count = count + 1;
        end

     end

end
```

Appendix 7.4 GUI of the ROI selection

1. Select a hypercube

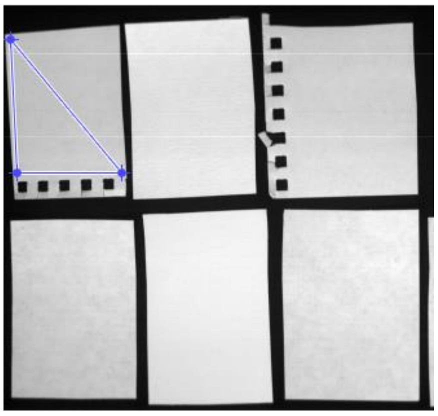


Select a .bip or a .bil file

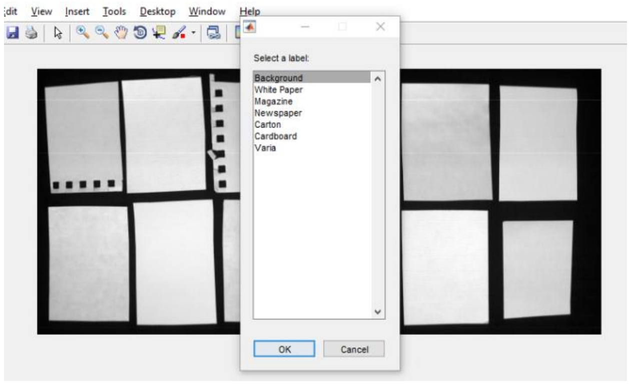
This PC > Desktop > NIR Data Resized > 15.04.16 NIR Tests > Group 1 - WP

Name	Date modified	Type	Size
WP_1-10	20.04.2016 17:50	BIP File	18 700 KB
WP_21-30	20.04.2016 18:08	BIP File	18 700 KB
WP_31-40	20.04.2016 18:08	BIP File	18 700 KB
WP_40_THICK	20.04.2016 18:08	BIP File	18 700 KB
WP_41-50	20.04.2016 18:08	BIP File	18 700 KB
Y_1-12	20.04.2016 18:08	BIP File	18 700 KB
Y_13-24	20.04.2016 18:08	BIP File	18 700 KB
Y_25-36	20.04.2016 18:08	BIP File	18 700 KB
Y_37-48	20.04.2016 18:08	BIP File	18 700 KB

2. Select area by drawing a ROI. Double-click to confirm



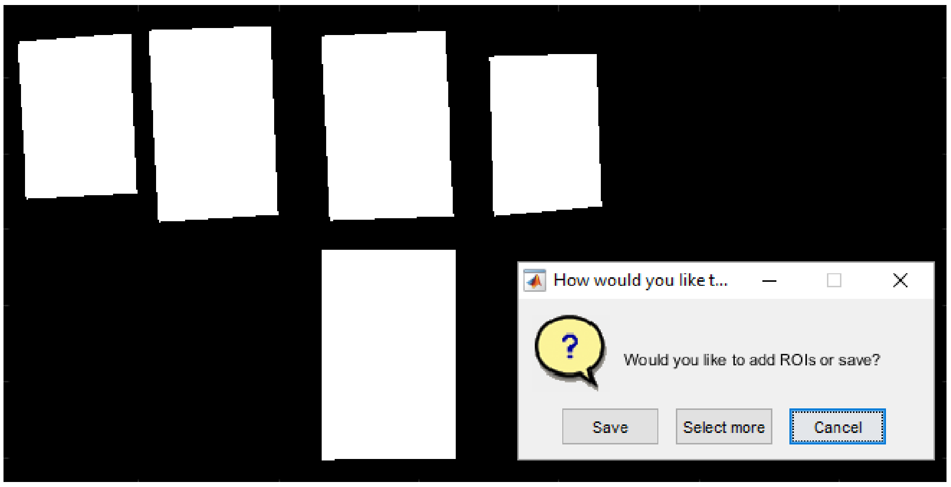
3. Select the class of the ROI



Select a label:

Background
White Paper
Magazine
Newspaper
Carton
Cardboard
Varia

4. Choose how to proceed:



How would you like t...

Would you like to add ROIs or save?

Appendix 7.5 `loadTrainingSpectra()`

```
% Get spectra from the ROIs for training data. The method of sampe
% selection can be selected in Section Selection
% Input:
%   path - the path of the scripts folder
% Output:
%   spectraOut -    n x 3 cell array of the extracted spectra, the
                    numerical label and the label

function spectraOut = loadTrainingSpectra(path)
    roiDirectory = uigetdir(path, 'Select directory of ROI files');
    roiDirectory = [roiDirectory, '\'];
    [spectraFileName,spectraPathName] = uiputfile('*.mat', 'Enter filename
where to save spectra');
    fullSpectraPath = fullfile(spectraPathName, spectraFileName);
    matFiles = dir(roiDirectory);

    % Read all the hypercubes that have a ROI file in the ROI directory
    for i = 1 : size(matFiles,1)
       if ~matFiles(i).isdir
           [hypercube, wavelengths, fileName] =
readHypercubeFromFileByName(strtok(matFiles(i).name, '.'), false, 4);
           disp([num2str(i), ' of ', num2str(size(matFiles,1))]);
           if isempty(hypercube)
               disp('Could not read hypercube');
           else
               hypercube = medfilt3(hypercube, [3 1 3]);

               %% Selection. Choose one of two options:
               % get a number of uniformly distributed data points of ROI
as
               % spectra:
               getNSamplesToPath(path, fileName, hypercube,fullSpectraPath,
roiDirectory);
               % get the medians of each ROI as spectra:
               %getMedianSpectraOfEachSampleToPath(path, fileName,
hypercube, fullSpectraPath, roiDirectory);
               %%
           end
       end
    end

    load(fullSpectraPath, 'spectra');
    if exist('spectra', 'var')
       spectraOut = spectra;
    else
       spectraOut = [];
    end
end
```

Appendix 7.6 `PLSDApredictHypercubeFromHierarchicalModel()`

```
% Uses the input model for classification and displays the pixel-by-pixel
% result
%
% Input:
%   modelIn - the PLS_Toolbox hierarchical classification model
%
% Outputs:
%   predictionData - the pixel-by-pixel
function [predictionData, hypercube, fileName] =
PLSDApredictHypercubeFromHierarchicalModel(modelIn)
    labelStrings = {'Not known', 'White paper', 'Magazine', 'Newspaper',
'Carton', 'Cardboard', 'WP D'};
                    %   0           1      2      3      4      5      6

    [hypercube, wavelengths, fileName] = readHypercubeFromFile(false, 40);
    hypercube = medfilt3(hypercube, [3, 1, 3]);
    data = reshape(hypercube, [size(hypercube, 1) * size(hypercube, 2),
size(hypercube, 3)]);

    prediction = modelIn.apply(data);
    predictionData = prediction.data;
    predictionData = reshape(predictionData, [size(hypercube, 1),
size(hypercube, 2), 1]);

    numberOfClasses = max(max(predictionData)) + 1;

    figure('Name', ['Predicted classes by the combined Hierarchical model,
', fileName])

    imagesc(predictionData)
    getColormapBasedOnClassNr(numberOfClasses, true)
    clrbar2 = colorbar;
    set(clrbar2,'YTick', ...
            0.5*(numberOfClasses - 1)/numberOfClasses:(numberOfClasses-
1)/numberOfClasses:numberOfClasses);
    set(clrbar2,'YTickLabel',labelStrings)
    axis equal
    axis([0,size(hypercube,2),0,size(hypercube,1)])
    xlabel('x - lines')
    ylabel('y - samples')
    title('Predicted classes by the PLS-DA model, strict classifictaion')

    figure('Name', 'Image of hypercube at central band')
    Y = squeeze(hypercube(:,:,ceil(size(hypercube,3)/2)));
    imagesc(Y)
    colormap(gray)
    axis equal

end
```

Appendix 7.7 `getGeneralizedPrediction()`

```
% Returns the generalized prediction from the pixel-by-pixel prediction
%
% Input:
%   predictionData - the pixel-by-pixel classification
%   hypercube - the hypercube
%
% Outputs:
%   generalizedPrediction - the generalized prediction mask/image
%   printPercentage -   an array with the print percentages. If No print or
%                       if not WP, then 0

function [generalizedPrediction, printPercentage] =
getGeneralizedPrediction(predictionData, hypercube)
    labelStrings = {'Not known', 'WP', 'MG', 'NP', 'CA', 'CB', 'WP D'};
                    %   0         1     2     3     4     5     6

    kmeans = kMeansClustering(hypercube, 3);
    kmeans = kmeans > 0;
    %figure; imshow(kmeans); title('Binarized k-means')

    kmeansMask = imfill(kmeans, 'holes');
    %figure; imshow(kmeansMask); title('fill holes')

    kmeansMask = bwareaopen(kmeansMask, 100);
    %figure; imshow(kmeansMask); title('Remove small blobs')

    maskConvHull = bwconvhull(kmeansMask, 'objects');
    %figure; imshow(maskConvHull); title('Convex hull')

    s = regionprops(maskConvHull, 'centroid','PixelIdxList');
    centroids = cat(1, s.Centroid);

    regionInfo = cell(size(s));
    printPercentage = zeros(size(s,1),1);

    generalizedPrediction = zeros(size(maskConvHull));
    for i = 1 : size(s, 1)
        regionGeneralization = zeros(size(maskConvHull));

        objectMask = false(size(maskConvHull));
        objectMask(s(i).PixelIdxList) = 1;

        regionGeneralization(objectMask > 0) = predictionData(objectMask >
0);
        regionLabels = predictionData(objectMask > 0);

        % find all labels in region:
        labels = unique(regionLabels);
        labelCounts = zeros(size(labels));
        for j = 1 : length(labels)
           labelCounts(j) = sum(regionLabels==labels(j));
        end
        [mostCount, mostIndex] = max(labelCounts);
        mostLabel = labels(mostIndex);
        areaSize = size(s(i).PixelIdxList,1);

        % If the most common label is zero, pick next common
        if mostLabel == 0
            labelCounts(mostIndex) = 0;
            [mostCount, mostIndex] = max(labelCounts);
            if mostCount / areaSize * 100 > 10
                mostLabel = labels(mostIndex);
            end
        end

        % Give precedence to NP in front of CB/CA
        if mostLabel == 4
            % if there's at least 10% of NP
            if labelCounts(4) / areaSize * 100 > 5
                labelCounts(mostIndex) = 0;
                [mostCount, mostIndex] = max(labelCounts);
                if labels(mostIndex) == 3
                    mostLabel = labels(mostIndex);
                end
            end
        end

        regionGeneralization(objectMask > 0) = mostLabel;
        regionInfo{i,1} = ['Pred: ', labelStrings(mostLabel + 1)];

        % Get print percentage by thresholding
        if mostLabel == 1
            Y =
mat2gray(squeeze(hypercube(:,:,ceil(size(hypercube,3)/2))));
            thresholdMask = im2bw(Y, 0.6);
            thresholdMask = thresholdMask & objectMask;
            whiteSize = nnz(thresholdMask);
            percentage = (1 - (whiteSize / areaSize)) * 100;
            printPercentage(i) = percentage;
            perText = [num2str(percentage, '%3.0f'), '% print'];
            regionInfo{i,1} = ['Pred:', labelStrings(mostLabel + 1),
perText];

        end

        generalizedPrediction = generalizedPrediction +
regionGeneralization;

    end

    numberOfClasses = max(unique(generalizedPrediction)) + 1;
    figure;imagesc(generalizedPrediction);
    hold on; plot(centroids(:,1),centroids(:,2), 'w*')
    hold off
    text(centroids(:,1), centroids(:,2) + 15, regionInfo,
'HorizontalAlignment', 'center', 'VerticalAlignment', 'top', 'Color', 'w')
    getColormapBasedOnClassNr(numberOfClasses, true)
    clrbar2 = colorbar;
    set(clrbar2,'YTick', ...
            0.5*(numberOfClasses - 1)/numberOfClasses:(numberOfClasses-
1)/numberOfClasses:numberOfClasses);
    set(clrbar2,'YTickLabel',labelStrings)
    title('Generalized classification')
    axis equal
    axis([0,size(hypercube,2),0,size(hypercube,1)])
end
```

Appendix 7.8 `validateModel()`

```
% User can select hypercube to validate. Also user has to provide the
folder
% of the ROI files (or modify the roiDirectory folder to point at the
% folder). Then the hypercube is classified using the hierarchical model,
% the generalized classification is given as well as errormaps for both.
%
% Input:
%   model - the hierarchical model to be used for classification
% Output:
%   predictionData - the pixel-by-pixel classification
%   printPercentage - an array of each sample's print percentage
%   pixelByPixelValidation -    the errormap of the pixel-by-pixel
%                               classification
%   generalizedValidation - the errormap of the generalized classification

function [predictionData, printPercentage, pixelByPixelValidation,
generalizedValidation] = validateModel(model)
    labelStrings = {'Not known', 'WP', 'MG', 'NP', 'CA', 'CB', 'WP D'};
                    %   0          1     2     3     4     5     6

    pixelByPixelValidation = zeros(0);
    generalizedValidation = zeros(0);

    [predictionData, hypercube, fileName] =
PLSDApredictHypercubeFromHierarchicalModel(model);

    [generalizedPrediction, printPercentage] =
getGeneralizedPrediction(predictionData, hypercube);

    % If the ROI files are in a constant location then it can be input here
    % In that case the uigetdir should be commented out
    %roiDirectory = 'C:\Users\Mari-Liis\Dropbox\MSc Thesis\MatLab
scripts\Validation set ROIs';
    roiDirectory = uigetdir(path, 'Select directory of Validation set ROI
files');
    roiDirectory = [roiDirectory, '\'];

    if exist([roiDirectory, fileName, '.mat'], 'file')
       load([roiDirectory, fileName, '.mat'], 'roiInfo');
       if exist('roiInfo', 'var')
           errorImage = zeros(size(predictionData));
           errorImageGen = zeros(size(predictionData));

           pixelByPixelValidation = zeros(size(roiInfo, 1),2);
           generalizedValidation = zeros(size(roiInfo, 1),2);

           centroidx = zeros(size(roiInfo, 1),1);
           centroidy = zeros(size(roiInfo, 1),1);

           regionInfo = cell(size(roiInfo, 1),1);

           for i = 1 : size(roiInfo, 1)
                errorPerROIimage = zeros(size(predictionData));
                mask = poly2mask(cell2mat(roiInfo(i,1))',
cell2mat(roiInfo(i,2))', size(hypercube,1), size(hypercube,2));
                centroidsROI = regionprops(mask, 'Centroid');
                centroidsROI = cat(1, centroidsROI.Centroid);
                centroidx(i) = centroidsROI(1,1);
                centroidy(i) = centroidsROI(1,2);
                mask = mask .* roiInfo{i,4};

                regionInfo{i,1} = ['Real: ', roiInfo{i,3}];

                % Get per-pixel errors
                errorPerROIimage(mask>0) = predictionData(mask>0);
                errorPerROIimage = errorPerROIimage - mask;
                errorPerROIimage = abs(errorPerROIimage) > 0;
                errorImage = errorImage + errorPerROIimage;
                ppErrorPercentage = (1 - nnz(errorPerROIimage) / nnz(mask))
* 100;
                ppTrueClass = roiInfo{i,4};
                pixelByPixelValidation(i,:) = [ppErrorPercentage,
ppTrueClass];

                % Get generalization error
                errorPerROIimage = zeros(size(predictionData));
                errorPerROIimage(mask>0) = generalizedPrediction(mask>0);
                errorPerROIimage = errorPerROIimage - mask;
                errorPerROIimage = abs(errorPerROIimage) > 0;
                errorImageGen = errorImageGen + errorPerROIimage;
                ppIsCorrect = nnz(errorPerROIimage) == 0;
                ppTrueClass = roiInfo{i,4};
                generalizedValidation(i,:) = [ppIsCorrect, ppTrueClass];

            end
            errorImage = errorImage > 0;
            figure;imagesc(errorImage);
             hold on; plot(centroidx,centroidy, 'k*')
             hold off
             text(centroidx, centroidy + 15, regionInfo,
'HorizontalAlignment', 'center', 'VerticalAlignment', 'top', 'Color', 'k')
             title(['Strict errormap for image ', fileName]);
             axis equal
             axis([0,size(hypercube,2),0,size(hypercube,1)])
             colormap([245/255 245/255 245/255; 90/255 180/255 172/255])

            figure;imagesc(errorImageGen);
             hold on; plot(centroidx,centroidy, 'k*')
             hold off
             text(centroidx, centroidy + 15, regionInfo,
'HorizontalAlignment', 'center', 'VerticalAlignment', 'top', 'Color', 'k')

             title(['Generalized errormap for image ', fileName]);
             axis equal
             axis([0,size(hypercube,2),0,size(hypercube,1)])
             colormap([245/255 245/255 245/255; 90/255 180/255 172/255])
        else
            disp('Could not find ROIs');
            pixelByPixelValidation = 0;
            generalizedValidation = 0;
        end
    else
        disp('Could not find ROI file for validation');
        pixelByPixelValidation = 0;
        generalizedValidation = 0;
    end
end
```